

日本国特許庁  
JAPAN PATENT OFFICE

24. 9. 2004

REC'D 23 DEC 2004

WIPO

PCT

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日 2003年 9月22日  
Date of Application:

出願番号 特願2003-330797  
Application Number:  
[ST. 10/C]: [JP 2003-330797]

出願人 カテナ株式会社  
Applicant(s):

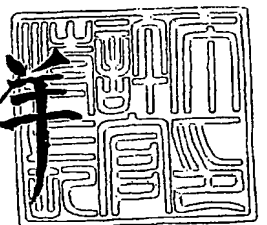
BEST AVAILABLE COPY

PRIORITY DOCUMENT  
SUBMITTED OR TRANSMITTED IN  
COMPLIANCE WITH  
RULE 17.1(a) OR (b)

2004年12月 9日

特許庁長官  
Commissioner,  
Japan Patent Office

小川 洋



【書類名】 特許願  
【整理番号】 DCT03035  
【提出日】 平成15年 9月22日  
【あて先】 特許庁長官殿  
【国際特許分類】 F01B 23/12  
F01C 17/00  
【発明者】  
【住所又は居所】 神奈川県鎌倉市十二所 9 6 7 - 6 4  
【氏名】 根来 文生  
【特許出願人】  
【識別番号】 598153401  
【氏名又は名称】 株式会社アイエスデー研究所  
【代表者】 根来 文生  
【特許出願人】  
【識別番号】 599086238  
【氏名又は名称】 ソフトウェア生産技術研究所株式会社  
【代表者】 根来 文生  
【代理人】  
【識別番号】 100110559  
【弁理士】  
【氏名又は名称】 友野 英三  
【手数料の表示】  
【予納台帳番号】 164782  
【納付金額】 21,000円  
【提出物件の目録】  
【物件名】 特許請求の範囲 1  
【物件名】 明細書 1  
【物件名】 図面 1  
【物件名】 要約書 1

## 【書類名】 特許請求の範囲

## 【請求項 1】

生産するソフトウェアが作動するコンピュータと人間とを介在するメディアに属する有意性の単位ごとに前記ソフトウェアの機能に関わらずに有意性を現実化する所与の普遍的構造を有しかかる構造に前記メディアに係る識別子が埋め込まれるべき第 1 の未定義部分及び該メディア上に存在する有意性獲得主体に係る識別子が埋め込まれるべき第 2 の未定義部分が含まれる第 1 の基軸プログラム（基底論理）と、

前記ソフトウェアの機能に関わらずに前記第 1 の基軸プログラムを複数の領域に展開するとともに前記第 1 の基軸プログラムのコンピュータ空間への演繹に際して補正を行う補正論理構造を必要に応じて展開する所与の普遍的な構造を有しかかる構造には前記第 1 及び第 2 の未定義部分が含まれる第 2 の基軸プログラム（パレット関数）と、

前記ソフトウェアの機能に関わらずに前記第 1 の基軸プログラム及び第 2 の基軸プログラムを前記有意性の単位及び前記メディアの分同期的に有意性が成立するように連鎖させる所与の普遍的な構造を有しかかる構造には前記第 1 及び第 2 の未定義部分が含まれる第 3 の基軸プログラム（パレット連鎖関数）と

の前記第 1 及び第 2 の未定義部分に前記ソフトウェアに係る開発要望からわり出したメディアに係る識別子及び該メディアに属する有意性獲得主体に係る識別子を代入することによりソフトウェアを一義的に決定することを特徴とするソフトウェアの生産方法。

【書類名】明細書

【発明の名称】ソフトウェアの生産方法

【技術分野】

【0001】

本発明は、ソフトウェア開発に係り、特に、数学的思考方にたったソフトウェア生産方法に関する。

【背景技術】

【0002】

#### 1. 1 ソフト世界の現世のジレンマ

ソフトウェアエンジニアリングの観点からこのジレンマを解明すれば、其の原因は以下の2点に集約されるであろう。

- 1 開発案件の曖昧性
- 2 開発案件を恣意的に論理化する事の焦り

上記1の問題は、誰もが作業を阻害する要因である事に異存はないであろう。他方上記2は、其れが作業を阻害する要因であるとの共通の認識であるかどうかは不明である。人により、其の事に意義を見出す場合もあるからである。

しかし、上記1及び2が原因となり生じるジレンマを以下の様に指摘する事が出来る。

- (01) ソフトの本質性を捉える技術規範が生まれにくい。
- (02) ソフト開発に携わる人々は当事者能力を失いやすい。
- (03) ソフト開発、並びに維持費用は効果と関係なく次第に高騰する。
- (04) ソフト開発の生産性は能力の低い管理者、能力の低い技術者のレベルに引き下げられやすい。
- (05) ソフトの品質を維持する技術規範が、管理規範に置きかえられ、技術其のものが軽んじられやすい。
- (06) 理不尽な見解が罷り通る。
- (07) 開発者が自分自身で作成したプログラムの正当性を自ら確信できない。
- (08) ソフトの市場価値が分かりにくい。
- (09) 生産コストの尺度が定まりにくい。

本研究では、上述の曖昧性や恣意性、並びにこれらに端を発するジレンマの問題は、われわれの意識に根差し、単に工学的問題として克服出来る性質のものではないと考える。この様な問題を解決する為には、ソフトを工学的視点だけではなく、意識の視点から捉え直す必要があると考える。

其れ故、本研究では、ソフトは意識世界で成立するとし、われわれの世界に其のまま移す事が出来ない存在であると考え。われわれに出来る事は、其の存在をプログラム言語を用いてプログラムとして述定する事だけであると考え。

其のプログラムについても、われわれはY2K騒動に見られる様な問題、プログラムが経時的に大きくなり、其れを構成する個々の問題を解決したとしても、其の全体の問題を解決出来ない合成の誤謬の問題を克服出来ない状況に遭遇している。

#### 1. 2 言表の限界

本研究では、われわれの言表行為を意識の世界を反映する手段であると考え。

しかし、ヴィトゲンシュタインの論考等によっても明らかな様に、其の言表行為には限界がある。



本研究では、この限界の問題を次の様に解釈する。

今、われわれが用いる事が出来る言葉の数を  $n$  個とする。今の時点におけるわれわれの言表行為は、其の  $n$  個以下の言葉を用いて行われる筈である。

他方、われわれの言表行為は未来に向かう時間の流れの中で逐次的に成立するので、われわれの言表行為は外延的に成立している。換言すれば、言表行為は  $n$  個の言葉を用いて未来を述定する行為となる。

他方、未来において、われわれが用いる事が出来る言葉の数は  $n$  個よりも大きくなっている筈である。換言すれば、其れを  $m$  個とすれば、 $m$  個の言葉の世界を  $n$  個の言葉で言表する事態がわれわれの外延的な言表行為である。本研究では、この言葉の数の差分が言表の限界、例えば、曖昧さを生じさせる原因であると考ええる。

われわれの外延的な言表行為は、時間的流れの中でどの程度の未来を述定しているのか、と言う問題もある。

この問題を是認する立場から、本研究ではわれわれが行う言表行為は、例えば過去と未来の一瞬の狭間の中で成立しているとの立場を取る。

本研究ではこの時間的な狭間を「同期」という言葉で表す。

【特許文献 1】国際公開第 97/16784 号パンフレット

【特許文献 2】国際公開第 98/19232 号パンフレット

【特許文献 3】国際公開第 99/49387 号パンフレット

【特許文献 4】国際公開第 00/79385 号パンフレット

【特許文献 5】国際公開第 02/42904 号パンフレット

【特許文献 6】特開 2002-202883 号公報

【発明の開示】

【発明が解決しようとする課題】

【0003】

われわれの外延的な言表行為は、時間的流れの中でどの程度の未来を述定しているのか、と言う問題もある。

この問題を是認する立場から、本研究ではわれわれが行う言表行為は、例えば過去と未来の一瞬の狭間の中で成立しているとの立場を取る。

本研究ではこの時間的な狭間を「同期」という言葉で表す。

本研究が目指す事は、結果的にこの同期の状態を述定する問題に置き換わるであろう。

【課題を解決するための手段】

【0004】

本研究では、意味とは同期状態において新たな存在が定義される作用の事だと定義する。其して、其の作用は意識の世界で成立すると考える。

本研究では、この作用をモデル化する。付言すれば、このモデルが本研究で言うソフトを定義する為の役割を果たす。

【発明の効果】

【0005】

本発明はソフトウェア産業のみならず、ソフトウェアを利用する全産業に極めて有用な効果をもたらす。

【発明を実施するための最良の形態】

【0006】

題目：ソフトウェア構築の摂理に関する統一理論の研究

Research on Methodological Universal Framework for Software Development

要旨

本研究の狙点は、ソフトをプログラム言語で述定する為に開発案件をアルゴリズム化する標準規則（公式）を求める事である。

例えば、われわれが言葉を発する直前迄、発する言葉を記憶する事が出来ないにも拘らず、われわれは言葉を無意識的に紡いで文章にしている。本研究では、これと同じ原理でプログラムが述定出来ないだろうかとして論考される。

この為に、本研究では「存在」「意図」「意識」に関する論考を形而上学的に行う。其して、其の結果、ソフトは「意識を成立させる外延的な存在を内包する作用」として定義される。其して、この定義から誘導される新たな定義により、プログラムを定義する公式が求められる。

本論文はこの研究に関する報告である。

#### Keywords：

意識を現す形而上学的存在，意識を捉えるアルゴリズム，ソフトを定義するアルゴリズム，プログラムを決定するアルゴリズム，プログラムを決定する公式。

#### 目次

#### 第一章 序論

- 1. 1 ソフト世界の現世のジレンマ
- 1. 2 言表の限界
- 1. 3 意味の生成
- 1. 4 本研究の論考の道筋

#### 第二章 本研究で用いられる概念

- 2. 1 基本概念
- 2. 2 仮説概念
- 2. 3 IDSに成立する定義の一覧

#### 第三章 ソフトの公式化モデル

- 3. 1 ENWの解説
- 3. 2 存在の考察
- 3. 3 存在の存在証明
- 3. 4 本仮説の適用
- 3. 5 PSの解説
- 3. 6 ベクトルの解説
- 3. 7 TDMの解説

#### 第四章 プログラムの公式

- 4. 1 SFの定義
- 4. 2 PRDの定義
- 4. 3 パレット連鎖関数の定義
- 4. 4 パレット関数の定義
- 4. 5 ベクトル種別の定義
- 4. 6 PRDの公式化
- 4. 7 始点の資格
- 4. 8 空概念のベクトルへの適用

- 4. 9 主語と座標の解説
- 4. 10 属性の解説
- 4. 11 ベクトルの定義規則
- 4. 12 プログラムの公式のスキーム

## 第五章 IDSに成立する定義の解説

### 第六章 纏め

- 6. 1 本研究の成果が齎すソフト開発作業上の効果
- 6. 2 本研究の成果が齎すソフト開発思想のパラダイム変革の可能性
- 6. 3 結論

## 図の説明

### 図1：IDSの模式図

IDSは、存在の基本概念を本研究で言う仮説概念を用いてモデル化したものである。図は、「IDS」「存在線」「不可知空間」「意識原子」「認識原子」「境界原子」「臨界自然数」「最小時間速度」「最大時間速度」「不可知空間の活力」の関係が示されている。本研究では、これらの定義からソフトの定義を成立させる為の新たな定義が求められる。

### 図2：論理原子の空間観念

時間速度はIDSに於いて論理原子として定義し直される。論理原子は空間観念を表象する。本研究では、空間観念は存在を表象する原理となる。

### 図3：占有空間

論理原子の集合が表象する空間的ひろがりの事で、集合に属す論理原子の空間観念の総和として定義する。

### 図4：空の連鎖

これは、「全体の性質」を有する内包的な存在を表象する。「意識原子」の集合（意識集合）が表す占有空間より、可能な限り近似的に小さな「空間観念」を有する「論理原子」が其の集合に属し、且つその論理原子の「複写の上限」を越えていなければ、其の論理原子を其の集合を「代表する空間」と定義する。この集合と代表空間の関係を「意識連鎖」と定義する。

### 図5：単元の連鎖

これは、「部分の性質」を有する外延的な存在を表象する。「認識原子」の集合が表す占有空間より、可能な限り近似的に大なる「空間観念」を有する「論理原子」が其の集合に属さない場合、その論理原子の「複写の上限」を越えていなければ、其の論理原子を其の集合を「代表する空間」と定義する。この集合と代表空間の関係を「認識連鎖」と定義する。

### 図6：連鎖の成立過程

本研究で用いる存在（連鎖）の成立過程を示す。

### 図7：臨界連鎖

意識連鎖の代表空間として「境界原子」が選ばれる場合、其の意識連鎖を臨界連鎖と定義する。

**図 8 : 特異連鎖**

認識連鎖の代表空間として「境界原子」が選ばれる場合、其の認識連鎖を特異連鎖と定義する。

**図 9 : 意図**

臨海連鎖と特異連鎖は、共に代表空間が境界原子となる事により等価な関係が成立する。本研究では、部分と全体が空間的に等しくなる事を意図の成立と定義する。

**図 10 : 意識**

臨界連鎖の履歴（意識  $\lambda$  集合）に属す意識原子数を  $m$  個とすれば、同じ数の認識原子を履歴（認

識  $\lambda$  集合）とする自然連鎖（含む群化自然連鎖）を、「意識を成立させる存在」と定義する。

**図 11 : ENW の模式図**

意識を成立させる存在（自然連鎖）から、「客体化」「陳述」により選ばれる自然連鎖（存在）が構成するネットワークを ENW と定義する。

**図 12 : ENW を構成する存在の関係**

ENW における存在は、「自分」「祖先」「子孫」の関係として定義する。

**図 13 : ENW の再定義**

ENW における存在の関係を述定可能にする為に、ENW の構造を再定義し直す。

**図 14 乃至 16 : PS の構造とその規則**

PS の構造は、7 種の述定文、並びに 4 種の領域で定義される。ENW の存在を外延的に述定させる為の作用を行う。本構造は述語構造として位置づけられ、「主語」を付与する事により命題

となる。其の命題を「ベクトル」と呼ぶ。PS は、「確立連鎖」「事象連鎖」「多重連鎖」の性質で三種の定義を成立させる。

**図 17 : TDM の模式図**

TDM は ENW の再定義されたモデルで、本研究で定義するソフトを表す。再定義の為に用いられる規則は、IDS に成立する定義である。1 個の ENW に対応して、1 個の TDM が定義される。

**図 18 : SF**


SF は、TDM をプログラム化する為のアルゴリズムのモデルである。其して、「パレット連鎖関数」「3 種のパレット関数」、並びに「パレットに属すベクトル」で定義される。

**図 19 : PRD**

複数の SF を 1 個のパレット連鎖関数で定義する構造を PRD と定義する。

**図 20 : パレット連鎖関数の定義規則**

パレット連鎖関数は、TDM の作用を内包的に述定したものである。複数の SF、パレットの作用を制御するプログラムとして定義される。



#### 図 2 1 及び 2 2：パレット関数の定義規則

パレット関数は、TDMの作用を外延的に述定したものである。パレットの種別ごとに定義され、ベクトルを制御する。

#### 図 2 3：I 2 の領域の関係

I 2 は、コンピューターの入力作用の役割を担うベクトルの事である。入力作用の方式を、PSの構造を基に形態的に定義した関係を示す。

#### 図 2 4：I 2 の作成単位

I 2 の定義単位は、論理体種別ごとである。

#### 図 2 5：O 4 の領域の関係

O 4 は、コンピューターの出力作用の役割を担うベクトルの事である。出力作用の方式を、PSの構造を基に形態的に定義した関係を示す。

#### 図 2 6：O 4 の作成単位

O 4 の定義単位は、論理体種別ごとである。

### 図表の説明

#### 図 2 7（表 1）：本理論の標準の骨格

これは、オブジェクト（主語）の種別、メソッド（ベクトルの作用）の種別、並びに両者に対応付けるアルゴリズムを定義する規則の関係表である。

#### 図 2 8（表 2）：主語の種別定義

開発案件に属す主語の種別の定義を示す。

#### 図 2 9（表 3）：主語の属性

主語に帯同する狭義の性質を本研究では属性と呼ぶ。

#### 図 3 0（表 4）：パレットに属すベクトルの種別

パレットに属すベクトル種別を示す。

#### 図 3 1（表 5）：S 4 の主語となる領域の種別

同期作用要素（S 4）が対象とする領域、即ち、同期作用要素を識別する主語である。

#### 図 3 2（表 6）：ベクトル種別と領域数

ベクトルに必要な領域の定義を示す。

#### 図 3 3（表 7）：L y e e B E L T

主語と其の履歴の関係を定義したものである。本研究は、履歴を属性と呼ぶ事がある。其して、履歴は狭義の属性と広義の属性から定義される。表 1 で示される属性とは、広義の属性の事である。

#### 図 3 4（表 8）：経路制御テーブル

これは、PRDの定義情報から再定義される。

#### 図 3 5 乃至 3 7（表 1 0）：ベクトルの型一覧表

これは、ベクトルの型の一覧表を示す。

図 38 (表 11) : L y e e B E L T の例

これは、表 7 を具体的に定義したものである。本例では、主語、並びに其の属性を定義する為の 42 項目、並びに表 12, 表 14, 表 15 で使用されるトークンが定義されている。

図 39 乃至 40 (表 12) : ベクトルの型の例

ベクトルを定義する為のテンプレートの総数 121 個の 1 例を示す。

図 41 乃至 42 (表 13) : ベクトルの例

表 12 のテンプレートから作り出されるベクトルを示す。

図 43 乃至 50 (表 14) : パレット関数の型の例

3 種のパレット関数を定義する為のテンプレートを示す。

図 51 乃至 65 (表 15) : パレット連鎖関数の型の例

パレット連鎖関数を定義する為のテンプレートを示す。

- 定義規則図 1 : 主語が正規でその属性が入力のベクトル
- 定義規則図 2 : 主語が正規でその属性が入力, 配列のベクトル
- 定義規則図 3 : 主語が正規でその属性が出力のベクトル
- 定義規則図 4 : 主語が正規でその属性が出力, 等価のベクトル
- 定義規則図 5 : 主語が正規でその属性が出力, 配列のベクトル
- 定義規則図 6 : 主語が正規でその属性が出力, 等価, 配列のベクトル
- 定義規則図 7 : 主語が正規でその属性が出力, 境界のベクトル
- 定義規則図 8 : 主語が正規でその属性が出力, 等価, 境界のベクトル
- 定義規則図 9 : 主語が正規でその属性が出力, 配列, 境界のベクトル
- 定義規則図 10 : 主語が正規でその属性が出力, 等価, 配列, 境界のベクトル
- 定義規則図 11 : 主語が K のベクトル
- 定義規則図 12 : 主語が K でその属性が等価のベクトル
- 定義規則図 13 : 主語が K でその属性が配列のベクトル
- 定義規則図 14 : 主語が K でその属性が等価, 配列のベクトル
- 定義規則図 15 : 主語が M でその属性が出力のベクトル
- 定義規則図 16 : 主語が M でその属性が出力, 等価のベクトル
- 定義規則図 17 : 主語が M でその属性が出力, 配列のベクトル
- 定義規則図 18 : 主語が M でその属性が出力, 等価, 配列のベクトル
- 定義規則図 19 : 主語が論理体でその属性が入力のベクトル
- 定義規則図 20 : 主語がアクセスキーでその属性が出力のベクトル
- 定義規則図 21 : 主語が処理条件キーでその属性が出力のベクトル
- 定義規則図 22 : 主語が論理体でその属性が出力のベクトル
- 定義規則図 23 : 主語がアクセスキーでその属性が出力のベクトル
- 定義規則図 24 : 主語が処理条件キーでその属性が出力のベクトル
- 定義規則図 25 : 主語がパレット W04 で R4 のベクトル
- 定義規則図 26 : 主語がパレット W02 で R2C のベクトル
- 定義規則図 27 : 主語がパレット W02 で R2 のベクトル
- 定義規則図 28 : 主語がパレット W03 で R3R のベクトル
- 定義規則図 29 : 主語が I2 第 2 領域の S4 のベクトル
- 定義規則図 30 : 主語が I2 第 4 領域の S4 のベクトル
- 定義規則図 31 : 主語が入力アクセスキーの L3 第 4 領域の S4 のベクトル

定義規則図 3 2 : 主語が入力処理条件キーの L 3 第 4 領域の S 4 のベクトル  
定義規則図 3 3 : 主語が O 4 第 4 領域の S 4 のベクトル  
定義規則図 3 4 : 主語が出力アクセスキーの L 3 第 4 領域の S 4 のベクトル  
定義規則図 3 5 : 主語が出力処理条件キーの L 3 第 4 領域の S 4 のベクトル  
定義規則図 3 6 : 主語が L 4 第 4 領域の S 4 のベクトル  
定義規則 3 7 から 4 7 については定義規則図 3 6 参照  
定義規則図 4 8 : 主語が L 2 第 4 領域の S 4 のベクトル  
定義規則 4 9 から 5 3 については定義規則図 4 8 参照  
定義規則図 5 4 : 主語が L 3 第 4 領域の S 4 のベクトル  
定義規則 5 5 から 6 9 については定義規則図 5 4 参照

A : 参考文献  
B : 本研究の国際学会発表論文一覧  
C : 本研究の講演一覧  
D : 本研究の第一回国際ワークショップ論文集  
E : 本研究で認められた特許一覧

#### 付録

これは、プログラム言語が VB の場合のベクトルの型、パレット関数の型、パレット連鎖関数の型の例である。

#### 発明の概要

##### 第一章 序論

本研究では、ソフトと「意識」は関わり合うとし、ソフトを定義する為には其の意識を述定する規則を求める事が必要だと考える。本論文は、この研究に関する報告である。

本研究では、其の基底に置かれるソフトは以下の 2 種のアルゴリズムで構成されると仮定する。即ち、

- 1 属人的に内在するアルゴリズム
- 2 公共的なアルゴリズム

其して、これら 2 種のアルゴリズムは、本研究では「意識を成立させる外延的な存在を内包する作用」として成立すると仮定する。本研究では、この作用を仮説世界を用いて定義する事を試みる。其して、この定義がソフトを定義する事に他ならないと考える。

この為、本研究では存在を形而上学的に定義し、其の存在を因に定義を発見する為の仮説世界を構築する。

本章では、ソフトウェア世界のジレンマ、ソフトウェア世界が遭遇している言表の限界の問題等に触れ、其の事に拠り本研究の意義を明らかにする。

#### 1. 1 ソフト世界の現世のジレンマ

ソフトウェアエンジニアリングの観点からこのジレンマを解明すれば、其の原因は以下の 2 点に集約されるであろう。

- 1 開発案件の曖昧性
- 2 開発案件を恣意的に論理化する事の焦り

上記 1 の問題は、誰もが作業を阻害する要因である事に異存はないであろう。

他方上記 2 は、其れが作業を阻害する要因であるとの共通の認識であるかどうかは不明である。人により、其の事に意義を見出す場合もあるからである。

しかし、上記 1 及び 2 が原因となり生じるジレンマを以下の様に指摘する事が出来る。

- (01) ソフトの本質性を捉える技術規範が生まれにくい。
- (02) ソフト開発に携わる人々は当事者能力を失いやすい。
- (03) ソフト開発、並びに維持費用は効果と関係なく次第に高騰する。
- (04) ソフト開発の生産性は能力の低い管理者、能力の低い技術者のレベルに引き下げられやすい。
- (05) ソフトの品質を維持する技術規範が、管理規範に置き換えられ、技術其のものが軽んじられやすい。
- (06) 理不尽な見解が罷り通る。
- (07) 開発者が自分自身で作成したプログラムの正当性を自ら確信できない。
- (08) ソフトの市場価値が分かりにくい。
- (09) 生産コストの尺度が定まりにくい。

本研究では、上述の曖昧性や恣意性、並びにこれらに端を発するジレンマの問題は、われわれの意識に根差し、単に工学的問題として克服出来る性質のものではないと考える。この様な問題を解決する為には、ソフトを工学的視点だけではなく、意識の視点から捉え直す必要があると考える。

其れ故、本研究では、ソフトは意識世界で成立するとし、われわれの世界に其のまま移す事が出来ない存在であると考え。われわれに出来る事は、其の存在をプログラム言語を用いてプログラムとして述定する事だけであると考え。

其のプログラムについても、われわれは Y 2 K 騒動に見られる様な問題、プログラムが経時的に大きくなり、其れを構成する個々の問題を解決したとしても、其の全体の問題を解決出来ない合成の誤謬の問題を克服出来ない状況に遭遇している。

## 1. 2 言表の限界

本研究では、われわれの言表行為を意識の世界を反映する手段であると考え。

しかし、ヴィトゲンシュタインの論考等によっても明らかな様に、其の言表行為には限界がある。

本研究では、この限界の問題を次の様に解釈する。

今、われわれが用いる事が出来る言葉の数を  $n$  個とする。今の時点におけるわれわれの言表行為は、其の  $n$  個以下の言葉を用いて行われる筈である。

他方、われわれの言表行為は未来に向かう時間の流れの中で逐次的に成立するので、われわれの言表行為は外延的に成立している。換言すれば、言表行為は  $n$  個の言葉を用いて未来を述定する行為となる。

他方、未来において、われわれが用いる事が出来る言葉の数は  $n$  個よりも大きくなっている筈である。換言すれば、其れを  $m$  個とすれば、 $m$  個の言葉の世界を  $n$  個の言葉で言表する事態がわれわれの外延的な言表行為である。本研究では、この言葉の数の差分が言表の限界、例えば、曖昧さを生じさせる原因であると考え。

われわれの外延的な言表行為は、時間的流れの中でどの程度の未来を述定しているのか、と言う問題もある。



この問題を是認する立場から、本研究ではわれわれが行う言表行為は、例えば過去と未来の一瞬の狭間の中で成立しているとの立場を取る。  
本研究ではこの時間的な狭間を「同期」という言葉で表す。

本研究が目指す事は、結果的にこの同期の状態を述定する問題に置き換わるであろう。

### 1. 3 意味の生成

本研究では、意味とは同期状態において新たな存在が定義される作用の事だと定義する。其して、其の作用は意識の世界で成立すると考える。  
本研究では、この作用をモデル化する。付言すれば、このモデルが本研究で言うソフトを定義する為の役割を果たす。

### 1. 4 本研究の論考の道筋

図 266 に、本研究の論考の道筋を示す。

本研究では、存在の基本概念（第二章 2. 1）を 5 個想定し、其れを公理とする。  
其して、この公理を本研究で想定される仮説概念（第二章 2. 2）7 個を用いてモデル化する。そのモデルを IDS と呼び、其の模式図は図 1 で示される。IDS で成立する定義の一覧表は第二章（2. 3）、其の詳細は第五章で解説される。

IDS の定義を用いて、ソフトを定義する為の 2 つのモデルが定義される。其れを PS, TDM と呼ぶ。PS, TDM をプログラム言語で述定可能な構造に定義し直したものをプログラムの公式と呼ぶ。

開発案件をプログラムの公式で述定する事により、開発案件を満たすソースプログラムが定義される。

## 第二章 本研究で用いられる概念

### 2. 1 基本概念

本研究で用いる存在の基本概念（公理）を以下に示す。

- (01) 履歴は複数の空間を要素とする空間である。
- (02) 履歴は 1 個の空間で代表（以下、代表空間）される。
- ...(03) 存在とは代表空間を有する履歴である。
- (04) 存在には外延的な性質を有する存在と内包的な性質を有する存在がある。
- (05) 外延的な存在の履歴に属す要素と内包的な存在の履歴に属す要素は、混在しない。

本研究では、存在の実体を代表空間、其の属性を履歴と呼ぶ。図 4、図 5 参照。

本研究では、外延的存在は部分を表し、内包的存在は全体を表す存在を意味する。其して、全ての外延的存在は内包的存在の影響を受けて定義される。われわれが感じられる存在は、外延的存在に限られる。しかし、われわれは内包的存在を知る事も感じる事も出来ない。

この定義から、例えば時間、精神、自分、林檎、路傍の石は、全て存在として同じ性質である。勿論、本研究の対象であるソフト、プログラムも同じ存在である。  
即ち、われわれの世界は、外延的存在を要素とする集合である。

## 2. 2 仮説概念

ここでは、存在の基本概念をモデル化する為に用いられる仮説概念を示す。

(01) 実数から無理数、循環少数、整数、並びに、後述する特異数を除く値の集合を  $U$  で表し、「不可知空間」と呼ぶ。存在するが  $U$  に属さない値を  $U$  の「特異数」と呼ぶ。

(02)  $U$  には部分集合が成立しない。

(03)  $U$  に属す値は其の値自身が自己作用を行う為の「自力」となり、「時間速度 ( $V_i$ )」と呼ぶ。

(04)  $U$  に属す時間速度の総和は  $U$  自身が自己作用を行う為の自力で、 $U$  の活力と呼ぶ。 $U$  の活力は  $\sum_1^\Phi V_i$  と表され、 $\Phi$  は時間速度の総数、共に定数で  $U$  の特異数である。

(05)  $U$  の活力の逆数は  $U$  自身が存在を示す際の値となり、 $U$  の占有空間と呼ぶ。 $U$  の占有空間を  $2\epsilon$  と表せば、 $\epsilon = 1 / 2 \sum_1^\Phi V_i$ 、 $\epsilon$  は定数で  $U$  の特異数である。 $2\epsilon$  は定数で最小の時間速度、其れ故、 $2\epsilon$  は  $U$  の特異数ではない。 $U$  に於ける時間速度の大小順序列を想定すれば、其の隣接する二つの時間速度の差分は  $2\epsilon$  である。

### 【0007】

本研究では、占有空間は集合体が有する体積的空間と解される。ここで以下の事に付言す

る。即ち、本研究では集合に属す1個の要素が有する体積的空間を空間観念と呼び、集合の空間観念を占有空間と呼ぶ。

### 【0008】

本研究の基本概念で言う代表空間とは、占有空間に近似する空間観念の事である。

(06)  $\Phi$  は定数で最大の自然数であると仮説し、 $\Phi = 2 \sum_1^\Phi V_i$  の関係が成立すると仮説する。 $\Phi$  を臨界自然数と呼ぶ。

(07)  $U$  に属す時間速度は絶える事なく1個ずつ無作為に選ばれ複写される。複写される時間速度は其の都度、 $U$  の外に  $U$  の作用として位相される。

## 2. 3 IDSに成立する定義の一覧

仮説概念を用いて存在の基本概念をモデル化する。ここでは其のモデルに成立する定義の名称を掲げる。定義の内容は第五章で説明する。

01: 境界時間速度 ( $V_B$ )

02: 時間速度 ( $V_i$ ) の位置活力

03:  $V_B$  の位置活力

04:  $V_i$  の空間観念

05: 境界時間速度の空間観念

06: 存在線

07: 論理原子

08: 不可知空間の着座座標

09: 論理原子の着座座標

10:  $\lambda$  集合

11: 占有空間

12: 意識  $\lambda$  集合の順列

13: 認識  $\lambda$  集合の順列

14: 空

15: 意識連鎖

16: 論理原子の複写回数の上限数

17: 単元

18: 確立連鎖

- 19：開示
- 20：連想
- 21：事象連鎖
- 22：多重連鎖
- 23：多重群化と自然群化
- 24：自然連鎖
- 25：認識連鎖
- 26：時間速度の割り込み
- 27：臨界連鎖
- 28：特異連鎖
- 29：意図
- 30：意識
- 31：客体化
- 32：陳述
- 33：客体化，陳述の停止
- 34：記憶と同化
- 35：転位と回帰

### 第三章 ソフトの公式化モデル

本章ではIDSの定義から導出される2つのモデル，即ち，述語構造モデル，並びに疑似的3次元空間モデルで定義されるソフトを定義する為のモデルについて述べる。前者をPS，後者をTDMで表す。

#### 3.1 ENWの解説

ENWとは，意識を成立させる存在（第五章：30参照）に作用する客体化（第五章：31参照），並びに陳述（第五章：32参照）によって，其の意識に至る群化履歴（参考文献A〔2〕参照）に属す自然連鎖が選ばれて構築されるネットワークの事である。この構造を本研究ではENWと呼ぶ。

図11で其の模式図を示す。ENWは，第一章（1.3）で述べる意識の中に成立するソフトのモデルを指している。

客体化は，意識を成立させる存在の履歴に属す空間観念が最小の認識原子（第五章：07参照）を選び，其れを代表空間とする自然連鎖を，意識を成立させる群化履歴の中から選ぶ。該当する自然連鎖が群化履歴の中に存在しなければ客体化は成立しない。

客体化で選ばれた自然連鎖の履歴に属す認識原子を代表空間とする自然連鎖を同じ群化履歴の中から選ぶ。この作用を陳述と言う。この場合，この群化履歴の中に該当する自然連鎖が存在すれば，選ぶ事が出来る限り自然連鎖を選ぶ。該当する自然連鎖が存在しなければ，自然連鎖は選ばれない。陳述により，選ばれる自然連鎖の履歴に属す認識原子を代表空間とする自然連鎖を，同じ群化履歴の中から選ぶ。この作用も陳述と呼ぶ。陳述は継続的に進められる。

客体化，並びに，陳述で選ばれる自然連鎖が多重連鎖（第五章：22参照）に遭遇すれば，其の客体化，或いは其の陳述は其処で停止する。結果的に，客体化，並びに陳述は，意識を成立させる存在を核とし，放射状に自然連鎖を解き放ち，更に其の自然連鎖を核とし，更なる自然連鎖を放射状に解き放つ。其して，それぞれの自然連鎖が多重連鎖に遭遇するまでこの放射状的な

構造を作り続ける。本研究では、この作用は自然空間（第五章：24参照）の中で絶える事なく継続して行われていると考える。

本研究では、E N Wは意識を表す存在の「崩壊」のパターンだと考える。其して、このパターンが意識を映し出す存在を成立させるのだと考える。  
本研究では、われわれが、例えば言葉を発するとはこのパターンを成立させている事だと考える。

本研究では、1個の意識を成立させる存在から成立するE N Wは時間的に瞬時に成立するとし、其の意味でE N Wにおける存在は同期していると考え。E N Wは、意識の存在が成立する数だけ成立するが、其れらは一斉に成立する事はない。其れ故、異なるE N Wの関係を本研究では非同期と呼ぶ。

因みに、第四章で言うS Fは、1個のS Fに対応して定義され、P R D複数のS Fに対応して定義されるものである。

本研究では、われわれが言う存在とはE N Wを構成する自然連鎖の事である。其れら存在は実体と属性で定義される。模式的には、図4、図5、図7、図8で示される構造である。

本研究で言う主語とは存在の実体を指し、本研究ではしばしば其れを「自分」として用いる。

本研究では、存在の属性を「履歴」と呼ぶ。主語の履歴には、広義の履歴と狭義の履歴がある。本研究ではL y e e B E L T（表7）として其れらは26項目で定義されている。表1で属性と定義されている項目は、広義の履歴の事である。後述するベクトルの定義規則で言う属性とは、広義の履歴を指す。

存在は、「記憶」を成立させる存在と、「同化」を成立させる存在（第五章：34参照）とに分けられる。本研究では、われわれが記憶出来る存在とは、上述の記憶を成立させる存在の事である。われわれが記憶できない存在とは、同化を成立させる存在の事である。

例えば、自分が林檎の形や味を知っていると言う事は、其の形や味は本研究ではそれぞれ自然連鎖であり、また、自分も自然連鎖で、其の自分が主になり、林檎の形や味の自然連鎖が従となり、自然群化（第五章：23参照）により自然連鎖が成立する事である。其して、其の自然連鎖が記憶の条件を満たす場合である。もし、其の自然連鎖が同化の条件を満たせば、自分は其の林檎の形や味に気付く事はない。

### 3.2 存在の考察

E N Wにおける意識を成立させる存在以外は、其の意識を成立させる存在の子孫と位置付けられる。換言すれば、全存在を子孫と位置付けられるのは意識を成立させる存在だけである。其の全存在は、更に祖先と子孫に分ける事が出来る。ここで言う祖先とは、子孫に先行して成立する存在の事である。子孫とは、直前の祖先によって選ばれる存在の事である。この立場に立てば、E N Wにおける自分は以下の3種の形態となる。

- (01) 自分は自分の祖先、並びに自分の子孫を有する。
- (02) 自分の祖先は不明であるが、自分は自分の子孫を有する。
- (03) 自分は自分の祖先を有するが、自分の子孫は不明である。図12参照。

### 3. 3 存在の存在証明

本研究では、自分の存在を証明する為に、以下の仮説を設ける。

即ち、自分の全祖先と1個の自分の子孫を自分で述語化させられれば、其の述語は自分の存在を証明しているとする。これをENWの基で言い換えれば、次の様になる。

#### (01) 証明の仮説 1

自分の全祖先と、自分の1個の子孫の述語化を自分で成立させられれば、其の述語は自分の存在を証明している。

#### (02) 証明の仮説 2

自分の全子孫の述語化が自分で成立させられれば、其の述語は自分の存在を証明している。

#### (03) 証明の仮説 3

自分の子孫の述語化が成立しなければ、其の述語は存在を証明する事にはならない。

#### (04) 証明の仮説 4

証明の仮説3の場合、証明の仮説1と2を用いて自分の存在を証明する事が出来る。

### 3. 4 本仮説の適用

上記の存在証明を、本仮説の事象連鎖、多重連鎖、確立連鎖を用いて述語化する。

(01) 事象連鎖は、其の定義から意識連鎖を述語化している。意識連鎖は、其の定義から全体を述語し、其の集合は全体の全体を述語している。其れ故、事象連鎖の述語は、其の事象連鎖に至る全ての事象連鎖を内包的に述語化している事に等しい。

(02) 多重連鎖は、其の定義から事象連鎖を外延的に述語化している事に等しい。

(03) 確立連鎖は、其の定義から其の確立連鎖に至る全ての存在を外延的に述語化している事に等しい。

上記(01)の述語は、自分の祖先に関する述語(L4)を表す。上記(02)の述語は、自分の子孫に関する述語(L3)を表す。上記(03)の述語は、既に成立している自分の子孫に関する述語(L2)を表す。

この場合、

(01) 上記3. 2 (01)の自分は、上記3. 3 (01)によりL4とL3で述語化される。

(02) 上記3. 2 (02)の自分は、上記3. 3 (02)によりL2で述語化される。

(03) 上記3. 2 (03)の自分は、上記3. 3 (03)により以下の様に述語化される。即ち、この場合の自分の祖先はL4で述語化可能である。他方、述語化すべき自分の子孫は不明なので、其の述語化は不能である。しかし、上記3. 3 (03)により、自分の子孫を述語化する必要がある。其の為に、述語化不能なL3はL4、L3、L2を成立させる別の存在を用いて置き換えられる。

### 3. 5 PSの解説

PSは、ENWにおける自分の存在証明を行う為に、自分で自分の述語化を行う為の構造(メソッド)である。

以下に、PSの構造、性質、並びに種別について述べる。

#### (01) PSの構造

PSは、図14で示されている様に7個のBOXで規約されている。それぞれを第

1 規約～第 7 規約と呼ぶ。また、4 種の領域は、第 2 領域、第 4 領域、第 6 領域、第 7 領域と呼ぶ。この構造に主語（自分）を付与することにより、其の主語の命題を成立させる述語が定義される。第 4 領域は、其の主語の命題の成立を立証する為に用いられる。

第 2 規約では、其の定義により、主語の命題の成否を図る為に主語に関する述語が定義される。其の成否の様子は、第 2 領域で一時的に留められる。

因みに、後述するベクトルでは、其の第 2 規約で定義される主語の述語は、プログラム言語の計算文か移送文となる。計算文の結果側の主語、移送文の結果側の主語が本研究で言う主語である。この主語を、本研究では端点と呼ぶ。

【0009】

其して、計算文の変数、移送文の起因となる変数は、始点と呼ぶ。通常、始点は主語に他ならない。

第 1 規約では、主語の命題化の成否を問う I F 文型の述語が定義される。その述語を成立させる為に用いる領域は第 4 領域である。結果として、この述語は主語の命題化の妥当性を確認する役割を担っている。

第 3 規約では、主語の述語化（第 2 規約）そのものの成否の様子を問う I F 文型の述語が定義される。その述語を成立させる為に用いる領域は、第 2 領域である。結果として、この述語は主語の命題化の検証性を確認する役割を担っている。

第 5 規約では、検証が不備な述語（第 2 規約）の内容を精査する I F 文型の述語が定義される。その述語を成立させる為に用いる領域は、全主語の第 4 領域の状態遷移である。

第 6 規約では、第 5 規約で発見される不備な述語に対処する為の述語が、第 6 領域を用いて定義される。第 7 規約では、第 5 規約で発見される不備な述語に対処する為の述語が、第 7 領域を用いて定義される。

第 6 規約と第 7 規約の述語の違いは、例えば第 2 規約の述語を成立させる祖先が検証不能の場合、第 6 規約の述語が定義され、検証が可能であるが存在を特定出来ない場合、第 7 規約の述語が定義される。

## (02) P S の性質

P S は、主語毎に固有の領域を有し、且つ、其れら領域が主語の性質に関係なく普遍的に規約される事において、それぞれの P S は独立して定義する事が出来る。即ち、P S は主語をオブジェクトとする普遍的なメソッドとなる。付言すれば、其れは最小のオブジェクトのクラスを定義する構造体である。

本研究で言うベクトルとは、このメソッドの事である。クラスをわれわれの認識を成立させる意味の単位であるとすれば、ベクトルは意味の最小単位を表す認識的構造と見る事が出来る。

ベクトルとなる場合の第 4 領域は、主語を表し、他の主語に開放される。

## (03) P S の種類

P S は、E N W に属す存在を定義する為に用いられる。

E N W に属す存在は、第三章（3. 2）で述べている様な種類がある。其れらを定義（述語化）する為に、本研究の仮説から得られる存在、即ち、事象連鎖、多重連鎖、確

立連鎖の性質を用いる。

【0010】

P S は、事象連鎖の性質で定義される P S、多重連鎖の性質で定義される P S、其して確立連鎖の性質で定義される P S に区別する事が出来る。

因みに、ベクトルについては後述するが、事象連鎖の性質で定義される P S は図 14-1 で表され、ベクトル化された場合、其れを L 4 で表す。確立連鎖の性質で定義される P S は図 14-2 で表され、ベクトル化された場合、其れを L 2 で表す。多重連鎖の性質で定義される P S は図 14-3 で表され、ベクトル化された場合、其れを L 3 で表す。

われわれの認識における開発案件に属す操作文は L 4、L 2、条件文は L 3 で扱われる。但し、この操作文、並びに条件文はベクトルを構成する 7 つの命令とは区別される。

本研究で定義されるプログラムの公式では、表 4 で示す様に、13 種のベクトルが定義される。其して、上述の 3 種と同様、他のベクトルも P S の構造を用いて定義される。詳細は第四章で述べる。

### 3.6 ベクトルの解説

P S に主語（自分）と其の履歴を付与する事により、其の P S は自分の存在を証明する為の述語を成立させる。本研究では、この述語をベクトルと呼ぶ。

ところでクラスとは、われわれが認識する対象を単位化する概念の一つと考える事が出来る。其して、其の定義は物質的、構造的、機能的、意味的等、様々な形態に適用する事が出来る。しかし、われわれは認識する対象を最小の単位（クラス）として定義出来るかどうかは不明である。例えば、名詞の樹木関係で其の末端の名詞が最小のクラスかどうかをわれわれは公共的に断定する事は出来ない。同様に、先頭の名詞が最大のクラスかどうかもまた、不明である。

ベクトルを定義する為の構造、即ち、P S は S F を構成する為の全ての種類のベクトルに共通となる。これは、ベクトルで定義される全てのクラスが同じ構造で定義される事である。換言すれば、この定義はどのクラスも最小化された場合に限り、成立するものである。其の事において、P S は、曖昧な対象をベクトルとして最小のクラスを決定する役割を果たすと考えられる。ベクトルは、必然的に各自独立し、後述するように S F の構成要素となる。其して、S F を動作させれば、ベクトル間に必然的な従属関係を成立させる。本研究ではこの作用を相補作用と呼ぶ。

其の事において、例えばオブジェクト指向設計で言うオブジェクトに関連付けるシーケンス図、アクティビティ図、コラボレーション図、データフロー図等で示される様な最終目的との関係において曖昧な規約しか提供する事が出来ないアイデアを、本研究では不要とする。

本研究によるプログラムの公式では、開発案件を満たすプログラムの論理は、相補作用によって生成される。換言すれば、其れは例えば 1 個のベクトルの命題の成立が、他の全ベクトルの命題を成立させる条件となり、また、他の全ベクトルの命題の成立が自分を成立させる為の条件となる事である。

これは、プログラムの論理の成立と存在の成立が同義で有る事を示唆し、且つ、論理も存在も全存在の基では無条件に成立し、部分の基では条件的にしか成立しない事が示唆するものである。

13種のベクトルの内、L2, L3, L4, 其して、R2Cのベクトルを第一種のベクトル、I2, O4のベクトルを第二種のベクトル、R4, R2, R3R, R3C, R3D, R3M, S4は第三種のベクトルと呼ぶ。

第一種のベクトルは、L y e e B E L T (表7) で定義される開発案件の情報と、其の定義規則で定義される。第二種のベクトルは、第一種のベクトル (入出力アクセスキー、入出力処理条件キー) と、其の定義規則から定義される。第三種のベクトルは、第一種、第二種のベクトル、S Fの構造規則 (図18), P R Dの構造規則 (図19) と、其の定義規則から定義される。

ベクトルの定義規則は第四章で説明される。

### 3.7 TDMの解説

本研究では、意図 (図9, 第五章:29参照) が成立し、其れにより意識 (図10, 第五章:30参照) が成立すると仮説している。其して、其の意識の成立を発端として、われわれが認識する存在、但し形而上学的存在が成立すると仮説している。以下に述べるTDM (図17) では、以上の関係が統一的に明示されている。

本研究では、ソフトは意識を成立させる存在の事であると考えられる。其して、其の定義 (述語化) を成立させる事が本研究の狙いである。しかし、其の存在は自分を成立させる起因であり、E N Wの構造の基では、其の存在を自分で定義 (述語化) する事は、自分で自分を定義する事と同じになり「同一律」に陥り、真偽の「偽」を正す事が不可能になってしまう。

本研究では、E N Wの構造を新たに定義し直し、且つ、其の構造の基でP Sの構造を用いて同一律に陥る事がない様に、上記3.3で述べた自分の存在証明を成立させる。この存在証明をE N Wに属す全存在について定義可能とする事により、あたかも其のE N Wを成立させる起因となる意識を成立させる存在の定義を成立するに等しいとする。この事により、ソフトの定義が成立すると考える。

新たに定義し直されたE N Wの構造を、TDMと呼ぶ。以下に、TDMの構造、並びに性質について述べる。

#### (01) TDMの構造

E N Wにおいて、任意の存在を自分とする。

其の場合、自分の祖先は自分により内包化 (過去) される関係に位置する。其して、自分の子孫は自分により外延化される関係に位置する。この場合、自分の祖先や子孫を定義する為の述語化は、例えば時間的に未来の存在を定義する事になってしまう。即ち、この事は子孫を定義する述語化は定義させられるが、祖先を定義する述語化は定義する事が出来ない事を意味する。

しかし、自分の存在証明を行う為には自分で自分の祖先を述語化する事が不可欠である。其の為に、祖先を子孫と同じ様に虚像化 (置換) させるシステム的な構造が必要になる。其れをE N Wに属す全存在について模式的に表せば、例えば図13である。これを更に抽象化し規約化すれば、其の様子は図17で示される。即ち、TDMである。

図13の左側に並ぶ存在 (3, 4, 5) は、E N Wの構造に従う存在である。これら存在は、外延的に定義可能となり、図16で述語化される。図13の右側に並ぶ存在 ((2), (3), (4)) は、E N Wに属す存在を虚像化したもので、図14で述語化される。



図13の存在(2)は、ENWの構造に従う存在で、図15で述語化される。図13の存在(1)は、ENWの構造を成立させる起因となる意識を成立させる存在で、述語化不能として除外される。

後述するプログラムの公式では、図14で述語化されるベクトルの集合は、記号W04、図15で述語化されるベクトルの集合は、記号W02、図16で述語化されるベクトルの集合は、記号W03で表され、パレットと総称される。因みに、パレットは主語をも兼ねる。パレットは、座標概念を有するベクトルの容器であり、後述するプログラムの公式では、パレット関数(図21及び22)として定義される。

TDM(図17)では、図13のL4で定義される存在は、虚像化されたものである。図17で示される存在は、われわれの認識上の存在を意味する。例えば、記憶される林檎のイメージは、本研究では図13で言う虚像化された存在、其して、より具体的に記憶される林檎のイメージは、例えば、われわれが見る(食べる)食卓の上にある林檎である。

TDMで定義されるパレットは、ENWを構成する存在に対し、TDMの擬似的空間の座標を与える役割を担っている。其の事により、L2、L3、L4がTDMの空間にわれわれが認識出来る存在を論理として定義する事が出来ると考える。

#### (02) TDMの性質

TDMは、1個の意識を成立させる存在について1個定義される。意識を成立させる存在は唯一つではないので、其の場合には、TDMは複数個定義される。其の関係は、PRDとして定義(次章)される。

TDMは、パレット関数を制御する関数として定義される。PRDの場合でも、其の関数は唯一つである。これをパレット連鎖関数と呼ぶ。パレット連鎖関数については、後述する。

### 第四章 プログラムの公式

ソフトは、意図や意識を強く反映して成立する存在の様に推測される事から、ソフトは形而上学的な存在として捉える事から始めた方が其の本質を解明する上で妥当性があるように思われる。

- 本研究はこの立場から、第二章、並びに第五章で述べる仮説を行い、其処に成立する定義から、ソフトは意識を成立させる存在として帰結される。しかし、われわれにはこの存在を言表する事は譬え仮説といえども不可能とすべきである。

其処で、本研究では意識を成立させる存在から客体化と陳述で選ばれる存在で構成されるENWを用いて、あたかも意識を成立させる存在を捉えた事に等しい述語構造を導き、其れをソフトの定義とするものである。

其の結果、本研究ではソフトは次の様に定義される。

即ち、「意識を成立させる存在を探索する作用」として位置付けられる。換言すれば、冒頭で述べた「意識を成立させる外延的な存在を内包する作用」と言う事になる。

本研究では、この作用をPSとTDMを利用して言表するものである。即ち、ソフトはPSとTDMを利用して言表される事になる。

これらの定義をプログラム言語で言表すれば、其れはプログラムである。付言すれば、ソフトを写すプログラムになる。この言表の作業が普遍的な規則で行えるならば、其の普遍的な規則はプログラムを定義する為の公式（以下、単にプログラムの公式と記す）となる。

以下に、ソフト化モデル（PS, TDM）を用いてプログラムを定義する公式について説明する。

#### 4.1 SFの定義

1個のTDMに呼応して定義されるプログラム言語で言表可能なTDMを、本研究ではSFと呼ぶ。SFの模式は図18で示す。TDMは、座標の集合体を律する空間でそれぞれの座標にベクトルが対応すると見る事が出来る。

#### 4.2 PRDの定義

複数のTDMに呼応して複数のSFをプログラム言語で言表する場合には、後述する経路作用要素（R2C, R3C, R3D, R3M）を用いれば、複数のSFを後述する1個のパレット連鎖関数で定義する事が出来る。この構造を、本研究ではPRDと呼ぶ。PRDは図19参照。

PRDが1個のSFで定義される場合もあるので、1個のSFを単に基本構造と呼ぶ。

因みに、主語が概算90万種のシステムを本方法で開発した例がある。其の時のPRDの定義枚数は、約1200枚、包含されるSFは約3000個であった。

TDMは、ベクトルが祖先を辿る作用を外延的に定義可能とする仕組を提供するのと同じ様に、PRDはSFが時系列的に未来から過去に溯る作用を外延的に成立させる仕組となる。開発案件の結論を表す命題は、例えば、図19のSF(1, 1)で定義される。其れを成立させる為の補助となるSFは、SF(2, 1)で与えられる。ここで言う補助とは、SF(1, 1)よりも以前に成立している存在の意味である。

PRDでは其の際の思考法とSFの性質の影響から以下の制限が規則として生じる。以下に、PRDを定義する場合のSFに関する制限事項を示す。

- (01) 直列するSFで隣り合うSFの上位のSFは、下位のSFと同期している。
- (02) 直列するSFで隣り合う下位のSFは、通常上位のSFに同期が成立する保証はない。
- (03) 直列する3個のSF間で、上位と最下位のSFの間には同期が成立する保証はない。
- (04) 並列するSFの間には、同期が成立する保証はない。

#### 4.3 パレット連鎖関数の定義

パレット連鎖関数は、図13で其の役割が示され、具体的には図20で定義されている。SFは、TDMの役割を具体的に定義したものである。パレットの同期状態、パレット間の同期状態、其して、異なるSF間の同期状態と其れらが成立させる非同期状態を統治する作用を行う。

特に、異なる S F 間を統治する為に作用する後述の 3 種の経路作用要素 (R 3 C, R 3 D, R 3 M) の役割を管理する。其の為に用いられる情報を、本研究では経路制御テーブル (表 8 参照) と呼ぶ。このテーブルは、P R D の情報が定義されれば決定する事が出来る。

#### 4. 4 パレット関数の定義

図 1 7 で示す座標、即ち図 1 3 で示すベクトルの集合は、S F ではパレットと呼ばれる。パレットは、ベクトルの集合と其れを統治する作用で定義される。この統治する作用をパレット関数と呼ぶ。

1 個の S F は 3 種のパレットから構成され、パレット関数も 3 種となる。それぞれ  $\Phi 4$ ,  $\Phi 2$ ,  $\Phi 3$  で示される。パレット関数の構造は図 2 1 及び 2 2 で示す。パレットを単に W 0 4, W 0 2, W 0 3 として表す事もある。

#### 4. 5 ベクトル種別の定義

パレットには、ベクトルが搭載される。搭載されるベクトルの種別は 10 種で、表 4, または図 2 1 及び 2 2 に其れらを示す。

他方、他にベクトルの概念と同じになる作用が 3 種あり、其れらはパレット連鎖関数の部分の作用として定義されている。図 2 0 参照。

ベクトル、は P S に主語を与える事により決定される。以下に 13 種のベクトルと、主語の関係を述べる。

##### (01) L 4

ベクトルの種別では、論理要素と呼ばれ、W 0 4 に搭載される。

この述語構造 (メソッド) は図 1 4 で示される。この述語構造に与えられる主語は、「正規単語, K 単語, M 単語」である。其して、これら主語は「出力, 等価, 配列, 境界」の属性を持つ。

主語の説明は表 2, 属性の説明は表 3 を参照。

##### (02) L 2

ベクトルの種別では、論理要素と呼ばれ、W 0 2 に搭載される。この述語構造 (メソッド) は図 1 5 で示される。この述語構造に与えられる主語は、「正規単語, K 単語」である。其して、これら主語は「入力, 配列」の属性を持つ。主語の説明は表 2, 属性の説明は表 3 を参照。

##### (03) L 3

ベクトルの種別では、論理要素と呼ばれ、W 0 3 に搭載される。この述語構造 (メソッド) は図 1 6 で示される。この述語構造に与えられる主語は、「正規単語, K 単語, M 単語, 入力アクセスキー, 入力処理条件キー, 出力アクセスキー, 出力処理条件キー」である。其して、これら主語は「出力, 等価, 配列」の属性を持つ。主語の説明は表 2, 属性の説明は表 3 を参照。

##### (04) I 2

ベクトルの種別では、入力作用要素と呼ばれ、W 0 2 に搭載される。この述語構造 (メソッド) は図 1 4 乃至 1 6 に準じて決定される。この述語構造に与えられる主語は、「論理体」である。其して、これら主語は「入力」の属性を持つ。論理体とは、例えば、画面, ファイル, 電文, 帳票などの単語 (主語) の集合体である。論理体の概念は、図 2 3, 並びにこのベクトルの作成単位は図 2 4 参照。因みに、L 2 の主語となる単語は、主

に I 2 の論理体を構成する項目である。

(05) O 4

ベクトルの種別では、出力作用要素と呼ばれ、W 0 4 に搭載される。この述語構造（メソッド）は図 1 4 乃至 1 6 に準じて決定される。この述語構造に与えられる主語は、「論理体」である。其して、これら主語は「出力」の属性を持つ。論理体とは、例えば、画面、ファイル、電文、帳票などの単語（主語）の集合体である。論理体の概念は、図 2 5、並びにこのベクトルの作成単位は図 2 6 参照。因みに、L 3、L 4 の主語となる単語は、主に O 4 の論理体を構成する項目である。

(06) R 4

ベクトルの種別では、経路作用要素と呼ばれ、W 0 4 に搭載される。この述語構造（メソッド）は図 1 4 乃至 1 6 に準じて決定される。この述語構造に与えられる主語は、「W 0 4」である。

R 4 は、W 0 4 を W 0 2 に接続する役割を果たすものである。換言すれば、R 4 は事象空間を確立空間に接続する役割になっている。即ち、事象空間が確立空間よりも先に作動しなければならない事を規約している。しかし、I D S の定義では、事象空間よりも先に確立空間が定義される。にも係わらず、S F では R 4 により逆の関係が定義される。これは、確立空間が事象空間に内包される本来の関係を、S F では逆転させている事を示す。其れは、事象空間は確立空間の外延（未来）として成立する空間であるにも係わらず、S F では事象空間が確立空間に内包（過去）される事になる。この作用は、R 4 とパレット連鎖関数で作り出される仕組みである。これは、L 4 が外延的に自分の祖先を述語する事が出来る様にする措置である。

後述する R 4 の定義規則は、簡潔な規則で定義される。しかし、本研究で言うソフトでは、R 4 は其れを成立させる本質的な役割を担っている。これは、R 4 が定義される事の重要な意義である。従来のプログラムに、この作用を見つけ出す事は出来ない。

(07) R 2

ベクトルの種別では、経路作用要素と呼ばれ、W 0 2 に搭載される。この述語構造（メソッド）は図 1 4 乃至 1 6 に準じて決定される。この述語構造に与えられる主語は、「W 0 2」である。

R 4 の意義と同じ立場で R 2 を見れば、R 2 は、確立空間と多重空間を接続する役割である。この接続の順位は R 4 の場合と異なり、I D S の定義に従っている。R 4 の様な特別な意義を有していない。しかし、其の事が意識を成立させる存在を捉える為の R 4 の措置を成立させる上では、欠かす事の出来ない役割を果たしている。

(08) R 2 C

ベクトルの種別では、経路作用要素と呼ばれ、W 0 2 に搭載される。この述語構造（メソッド）は図 1 4 に準じて決定される。この述語構造に与えられる主語は、「W 0 2」である。このベクトルは、自分が属す S F 以外の隣接する下位の S F の W 0 4 と接続する。このベクトルの意義は、R 2 と同じである。

(09) R 3 R

ベクトルの種別では、経路作用要素と呼ばれ、W 0 3 に搭載される。この述語構造（メソッド）は図 1 4 乃至 1 6 に準じて決定される。この述語構造に与えられる主語は、「W 0 3」である。このベクトルの意義は、R 4 と同じである。

## (10) R3C

ベクトルの種別では、経路作用要素の役割を果たし、パレット連鎖関数で定義される。この作用の役割の意義は、R4と同じで、且つ、接続する空間は異なるSFの空間（事象空間）である。其の為、この作用はパレットの上位に当たるパレット連鎖関数で統治される。この述語構造は、パレット連鎖関数（図20）の部分として其の基で定義される。パレット連鎖関数で統治される経路作用要素は、主語を持たない。

## (11) R3D

ベクトルの種別では、経路作用要素の役割を果たし、パレット連鎖関数で定義される。この作用の役割の意義は、これまでの経路作用要素とは異なる。即ち、異なるSFで下位のSFが隣接する上位のSFの同じ空間（多重空間）を接続するものである。異なるSFの空間を接続するという事において、この作用はパレットの上位に当たるパレット連鎖関数で統治される。この述語構造は、パレット連鎖関数（図20）の部分として

其の基で定義される。パレット連鎖関数で統治される経路作用要素は、主語を持たない。

## (12) R3M

ベクトルの種別では、経路作用要素の役割を果たし、パレット連鎖関数で定義される。この作用の役割の意義は、R3に準じている。しかし、異なるSF間の空間を接続するものである。即ち、下位のSFのW03（多重空間）と間接上位のSFのW04（事象空間）を接続する。しかし、異なるSFの空間を接続するという事において、この作用はパレットの上位に当たるパレット連鎖関数で統治される。この述語構造は、パレット連鎖関数（図20）の部分として其の基で定義される。パレット連鎖関数で統治される経路作用要素は、主語を持たない。

## (13) S4

ベクトルの種別では、同期作用要素と呼ばれ、W04に搭載される。この述語構造（メソッド）は図14乃至16に準じて決定される。この述語構造に与えられる主語は「領域」で、其の詳細は表5で示されている。

本研究で言うソフトは、論理的な様相を定義するこれまでのプログラムと異なる。これまでのプログラムは、機能を実現させる為に其の述語化をステートメントと其の順序性を用いて達成している。この場合には、ステートメントと順序性の関係が重要な課題となる。

本研究で言うプログラムは、機能を実現させる為に其の述語化が行われるのではない。本研究で言うプログラムを作動させる事によって、其の機能が得られる様になっている。これは、本研究で言うソフトの捉え方がENWで述べている様に、機能ではなく、同期構造を捉える仕組みになっているからである。しかし、われわれがプログラムを定義する限りにおいてプログラムに求めるものは機能である。其して、其の機能は同期状態に対し其れらが遷移する様相、即ち、非同期の構造の基で成立するものである。

同期作用要素は、出力作用要素と共に同期状態を非同期の状態に換える役割を果たすベクトルである。具体的には、領域の状態を用いて其れを成立させる。例えば、特定の領域のデータの存在の有無を制御する事によって、其れを実現させる。この場合、同期作用要素の作用は、SF、またはPRDの構造を用いて定義される事にならなければならない。定義規則では其の様になっている。

ベクトルの主語、作用、定義規則（後述）の関係は、表1で示されている。

本研究では、主語は既述の E N W (第三章) で言う存在に該当し、所謂、業務開発案件を構成する最小の論理性を宣言する為のものである。

#### 4. 6 P R D の公式化

P R D を定義する公式を以下の様に示す。

(01) パレットを次の様に定義する。

$W 0 4 = \Phi 4$  (W 0 4 に属すベクトル)

$W 0 2 = \Phi 2$  (W 0 2 に属すベクトル)

$W 0 3 = \Phi 3$  (W 0 3 に属すベクトル)

(02) S F を次の様に定義する。

$S F = \Phi 0 (W 0 4 + W 0 2 + W 0 3)$

図 1 8 参照。

(03) P R D を次の様に定義する。

$P R D = S F$  の集合

図 1 9 参照。

#### 4. 7 始点の資格

主語は、其の論理性を P S の第 2 規約で宣言する。其の場合に、端点として他の主語を用いる。其の場合の始点となる主語の資格を以下に述べる。

(01) W 0 4 に属す主語は、同じ W 0 4 の他の主語を始点として利用する事が出来る。

(02) W 0 4 に属す主語は、同じ S F の W 0 2 の主語を始点として利用する事が出来る。

(03) W 0 4 に属す主語は、隣下位の S F の W 0 4 の主語を始点として利用する事が出来る。

(04) W 0 4 に属す主語は、隣下位の S F の W 0 2 の主語を始点として利用する事が出来ない。

(05) W 0 4 に属す主語は、同位の他の S F の W 0 4 の主語を始点として利用する事が出来ない。

(06) W 0 4 に属す主語は、隣隣下位の S F の W 0 4 の主語を始点として利用する事が出来ない。

(07) 上記 (06) で隣隣下位の S F の W 0 4 の主語を始点として利用したい場合には、隣隣下位の S F の W 0 4 の主語のベクトルは其の第 4 領域と同じ領域を隣隣上位の W 0 4 に設け、且つ、其の領域に自分の第 4 領域と同じ状態を移送する。隣隣上位の W 0 4 に属す主語は、其の領域を始点として利用する事が出来る。この措置は、I D S では転位 (第五章: 35 参照) と呼ばれる定義に準じている。

(08) W 0 4 に属す主語は、隣上位の S F の W 0 4 の主語を始点として利用する事が出来ない。

(09) 上記 (08) で隣上位の S F の W 0 4 の主語を始点として利用したい場合には、上記 (07) と同じ形式の措置を取る。隣下位の W 0 4 に属す主語は、其の領域を始点として利用することが出来る。この措置は、I D S では回帰 (第五章: 35 参照) と呼ばれる定義に準じている。

#### 4. 8 空概念のベクトルへの適用

本研究では、空 (図 4, 第五章: 14 参照), 単元 (図 5, 第五章: 17 参照) の概念が定義

されている。定義される存在が部分を示す存在の場合には、其の存在は「単位」で定義される。全体を示す存在の場合には、「空」で定義される。

本研究では、存在は「連鎖」で述定される。連鎖には、既に述べている様に5種の連鎖（意識連鎖、確立連鎖、事象連鎖、多重連鎖（含む群化多重連鎖）、自然連鎖（含む群化自然連鎖））がある。図4は意識連鎖で、図5は其れ以外の連鎖で、其の模式的である。意識連鎖は全体を指す存在、他の連鎖は部分を指す存在である。換言すれば、全体を指す存在は、自分で自分を述語化する事が出来る性質の存在である。部分を指す存在は、自分で自分を述語化する事が出来ない性質の存在である。

因みに、ENWに属す存在は、後者の存在である。其の為に、後者の存在は第三章（3.3）で述べている様に、祖先や子孫を用いて自分の述語化を成立させるのである。其の為に、単元の性質を持つ存在を、この空の概念を利用して、其の述語化が成立する様に述語構造（PS）が定義されるのである。

ここで、この立場から第三章（3.5）で述べたPSの構造について付言する。図14のPSの主語（自分）は、祖先の存在を確認する述語を成立させなければならない。しかし、祖先は自分の存在より以前に成立しており、自分の祖先が成立する時点に戻す事は不可能である。其して、自分は自分の祖先を自分の存在よりも先に成立するかのようにならなければ、述語化することが出来ない。この事は、第三章（3.4, 3.7）と第四章の冒頭の既述を言い換えるものである。其の為に述語化の構造が必要となり、其れがPSである。其のPSは、祖先をあたかも子孫であるかのように述語されるPSの第2規約の内容を、子孫ではなく祖先としての内容に置き換える役割を果たさなければならない。其の為に、空の概念が用いられる。

PSの第1規約の述語は、其れがベクトルに置き換えられる場合、ベクトルの第4領域の内容の有無（意味ではない）を問う。もし、第4領域の内容が無であれば、其のベクトルの第2規約で自分の祖先が外延的に述語化される。有であれば、其の述語化は行われない。

「無」とは、自分の祖先がまだ述語化されていない事を意味する。ベクトルではこれを「非空」として記す。「有」とは、自分の祖先が既に述語化されている事を意味する。ベクトルではこれを「空」として記す。

もし、自分が意識連鎖で定義される存在であれば、自分は自分の祖先を自分で述語する事が出来る。しかし、自分は単位で定義される存在である。其の理由により、自分は自分の祖先を述語する為には自分の祖先を自分の履歴の中に存在させる必要がある。

本研究で言う内包化とは、この意味の事である。其して、これを実現させる為に、ベクトルは自分の履歴に属す存在を自分を含む全主語として成立する様に構造的に成立させて、其の基で自分の第1規約で自分の第4領域の内容の有無を問う形態になっている。これは、擬似的に意識連鎖を成立させている事である。其の事によって、述語化不可能な祖先を可逆化させて述語化を可能にさせるものである。

#### 4.9 主語と座標の解説

パレット種別、SF, PRD, 主語を要素とする集合体（本研究では論理体、或いは定義体などと呼ぶ、例：画面、帳票、ファイル）は、主語を特定する為に主語に付与される情報となる。これを本研究では主語の座標と呼ぶ。ベクトルが主語で定義される事を想起すれば、其のベクトルは其の主語により座標を有する事になる。この概念は、主語を一義化

する為に用いられる。例えば、同義語はこの座標によって一義化される。因みに、一義化されない場合には、後述する「等価」の属性により区別化される。

この座標の概念は全主語に付与されるので、既述の始点となる主語も座標を有する事になる。其の事によって、既述の相補作用が自律的に成立する事になる。

#### 4. 1 0 属性の解説

主語には 2 2 の種類があるが、属性を有するものと有しないものとがある。属性を有しないものは、経路作用要素の主語となるパレットだけである。パレットは、SF を決定する基底情報で普遍性が与えられているからである。属性は、5 種類（入力、出力、配列、等価、境界）あり、その定義は表 3 参照。

#### 4. 1 1 ベクトルの定義規則

ベクトルの定義規則は、定義規則図として示す。定義規則図に現れる共通の事柄は、前以て以下に示す。

##### (01) ベクトルの表記法

ベクトルの表記は、ベクトル種別、主語、属性の順に定義されている。以下に例を示す。

【0011】

例) L 2 正規 (入力・配列)

上例の L 2 はベクトル種別、正規は主語、入力並びに配列は属性を表す。

##### (02) ベクトル種別が I 2 の第 1 規約の表記の説明

###### ・表記

I 2 第 2 ≠ 空 AND 入力アクセスキー L 3 第 4 = 空  
AND 入力処理条件キー  
L 3 第 4 = 空

###### ・表記の意味

I 2 の第 2 領域にデータがない事、且つ、入力アクセスキーの L 3 第 4 領域にデータがある事、且つ、入力処理条件キーの L 3 第 4 領域にデータがある事 の判定を意味する。

##### (03) ベクトル種別が L 2 の第 1 規約の表記の説明

###### ・表記

L 2 第 4 領域 ≠ 空

###### ・表記の意味

L 2 の第 4 領域にデータがない事の判定を意味する。

##### (04) ベクトル種別が L 3 で主語が単語の第 1 規約の表記の説明

###### ・表記

L 3 第 4 領域 ≠ 空

###### ・表記の意味

L 3 の第 4 領域にデータがない事の判定を意味する。

##### (05) ベクトル種別が L 3 で主語が入力アクセスキー、入力処理条件キー、出力アクセスキー、出力処理条件キーの第 1 規約の表記の説明

###### ・表記



NOP

- ・表記の意味  
無条件に第 2 規約に進む事を意味する。

(06) ベクトル種別が L 4 の第 1 規約の表記の説明

- ・表記  
L 3 第 4 領域 = 空  
AND L 4 第 4 領域 ≠ 空
- ・表記の意味  
L 3 の第 4 領域にデータがある事、且つ、L 4 の第 4 領域にデータがない事の判定を意味する。

(07) ベクトル種別が L 4 で属性が等価の第 1 規約の表記の説明

- ・表記  
L 3 第 4 領域 = 1  
AND L 4 第 4 領域 ≠ 空
- ・表記の意味  
L 3 の第 4 領域のデータが 1 である事、且つ、L 4 の第 4 領域にデータがない事の判定を意味する。等価数を 2 と置いた時の例を定義規則図は示しているが、1 つ目を 1 と定めた事を意味する。

(08) ベクトル種別が L 4 で属性が等価の第 1 規約の表記の説明

- ・表記  
L 3 第 4 領域 = 2  
AND L 4 第 4 領域 ≠ 空
- ・表記の意味  
L 3 の第 4 領域のデータが 1 である事、且つ、L 4 の第 4 領域にデータがない事の判定を意味する。等価数を 2 と置いた時の例を定義規則図は示しているが、2 つ目を 2 と定めた事を意味する。

(09) ベクトル種別が L 4 で K の派生元となった正規の第 1 規約の表記の説明

- ・表記  
L 3 第 4 領域 = 空  
AND L 4 第 2 領域 ≠ 空
- ・表記の意味  
L 3 の第 4 領域にデータがある事、且つ、L 4 の第 2 領域にデータがない事の判定を意味する。

(10) ベクトル種別が O 4 の第 1 規約の表記の説明

- ・表記  
O 4 第 4 の出力済フラグ ≠ 空  
AND 出力アクセスキー L 3 第 4 = 空  
AND 出力処理条件キー L 3 第 4 = 空  
AND 経路制御 T B L の自 S F の R 3 D = 発報済
- ・表記の意味  
O 4 の第 4 領域のステータス状態を示す出力済フラグにデータがない事、且つ、出力アクセスキーの L 3 の第 4 領域にデータがある事、且つ、出力処理条件キーの L 3 の第 4 領域にデータがある事、且つ、経路情報 T B L の自 S F の R 3 D が発報済みである事の判定を意味する。

- (11) ベクトル種別が R 4 の第 1 規約の表記の説明
- ・表記  
R 4 第 4 領域 ≠ 空
  - ・表記の意味  
R 4 の第 4 領域にデータがない事の判定を意味する。
- (12) ベクトル種別が R 2 C の第 1 規約の表記の説明
- ・表記  
R 2 C 第 4 領域 ≠ 空
  - ・表記の意味  
R 2 C の第 4 領域にデータがない事の判定を意味する。
- (13) ベクトル種別が R 2 の第 1 規約の表記の説明
- ・表記  
R 2 第 4 領域 ≠ 空
  - ・表記の意味  
R 2 の第 4 領域にデータがない事の判定を意味する。
- (14) ベクトル種別が R 3 R の第 1 規約の表記の説明
- ・表記  
R 3 R 第 4 領域 ≠ 空 AND  
(自 S F L 4 今回状態変化フラグ ≠ 0  
OR 自 S F L 3 今回状態変化フラグ ≠ 0  
OR 自 S F L 2 今回状態変化フラグ ≠ 0)
  - ・表記の意味  
R 3 R の第 4 領域にデータがない事、且つ、自分の S F の L 4 今回状態変化フラグが 0 でない事、または、自分の S F の L 3 今回状態変化フラグが 0 でない事、または、自分の S F の L 2 今回状態変化フラグが 0 でない事のいずれかが成立した事の判定を意味する。
- (15) ベクトル種別が S 4 で主語が I 2 第 2 領域の第 1 規約の表記の説明
- ・表記  
L 2 第 4 領域 ≠ 空
  - ・表記の意味  
L 2 の第 4 領域にデータがない事の判定を意味する。
- (16) ベクトル種別が S 4 で主語が I 2 第 4 領域／単語の L 3 第 4 領域／O 4 の第 4 領域の第 1 規約の表記の説明
- ・表記  
NOP
  - ・表記の意味  
無条件に第 2 規約に進む事を意味する。
- (17) ベクトル種別が S 4 で主語が入力アクセスキーの L 3 第 4 領域／入力処理条件キーの L 3 第 4 領域／出力アクセスキーの L 3 第 4 領域／出力処理条件キーの L 3 第 4 領域／L 4 第 4 領域の第 1 規約の表記の説明
- ・表記  
O 4 第 4 の出力済フラグ = 空
  - ・表記の意味  
O 4 第 4 の出力済フラグにデータがある事の判定を意味する。

- (18) ベクトル種別が S 4 で主語が L 2 第 4 領域の第 1 規約の表記の説明
- ・ 表記  
出力処理条件キーの L 3 第 6 領域 = 空
  - ・ 表記の意味  
出力処理条件キーの L 3 第 6 領域にデータがある事の判定を意味する。
- (19) ベクトル種別が I 2 の第 2 規約の表記の説明
- ・ 表記  
入力コマンド文
  - ・ 表記の意味  
入力コマンドの命令を意味する。
- (20) ベクトル種別が L 2 の第 2 規約の表記の説明
- ・ 表記  
入力論理体に属す主語のデータ → L 2 第 2 領域
  - ・ 表記の意味  
入力論理体に属す主語のデータを, L 2 の第 2 領域に移送する事を意味する。
- (21) ベクトル種別が L 3 の第 2 規約の表記の説明
- ・ 表記  
N O P
  - ・ 表記の意味  
ノーオペレーションを意味する。
- (22) ベクトル種別が L 4 の第 2 規約の表記の説明
- ・ 表記  
主語の論理性 → L 4 第 2 領域
  - ・ 表記の意味  
主語の論理性とは業務要件を指す。移送命令や計算式の結果のデータを, L 4 の第 2 領域に移送する事を意味する。
- (23) ベクトル種別が L 4 で属性が等価の第 2 規約の表記の説明
- ・ 表記  
主語の論理性 A → L 4 第 2 領域
  - ・ 表記の意味  
主語の論理性とは業務要件を指す。移送命令や計算式の結果のデータを, L 4 の第 2 領域に移送する事を意味する。等価数を 2 と置いた時の例を定義規則図は示しているが, 1 つ目の業務要件を A と定めた。
- (24) ベクトル種別が L 4 で属性が等価の第 2 規約の表記の説明
- ・ 表記  
主語の論理性 B → L 4 第 2 領域
  - ・ 表記の意味  
主語の論理性とは業務要件を指す。移送命令や計算式の結果のデータを, L 4 の第 2 領域に移送する事を意味する。等価数を 2 と置いた時の例を定義規則図は示しているが, 2 つ目の業務要件を B と定めた。
- (25) ベクトル種別が L 4 で K 主語の第 2 規約の表記の説明
- ・ 表記

派生元の論理性→L 4 第 2 領域

- ・ 表記の意味

派生元の論理性とは、Kの派生元となった正規主語の業務要件を指す。移送命令や計算式の結果のデータを、L 4 の第 2 領域に移送する事を意味する。

(26) ベクトル種別がL 4 でK主語で属性が等価の第 2 規約の表記の説明

- ・ 表記

派生元の論理性A→L 4 第 2 領域

- ・ 表記の意味

派生元の論理性とは、Kの派生元となった正規主語の業務要件を指す。移送命令や計算式の結果のデータを、L 4 の第 2 領域に移送する事を意味する。等価数を 2 と置いた時の例を定義規則図は示しているが、1つ目の業務要件をAと定めた。

(27) ベクトル種別がL 4 でK主語で属性が等価の第 2 規約の表記の説明

- ・ 表記

派生元の論理性B→L 4 第 2 領域

- ・ 表記の意味

派生元の論理性とは、Kの派生元となった正規主語の業務要件を指す。移送命令や計算式の結果のデータを、L 4 の第 2 領域に移送する事を意味する。等価数を 2 と置いた時の例を定義規則図は示しているが、2つ目の業務要件をBと定めた。

(28) ベクトル種別がL 4 でK主語の派生元となった正規主語の第 2 規約の表記の説明

- ・ 表記

KのL 2 第 4 領域→L 4 第 2 領域

- ・ 表記の意味

KのL 2 の第 4 領域のデータを、L 4 の第 2 領域に移送する事を意味する。

(29) ベクトル種別がO 4 の第 2 規約の表記の説明

- ・ 表記

出力コマンド文

- ・ 表記の意味

出力コマンドの命令を意味する。

(30) ベクトル種別がR 4 の第 2 規約の表記の説明

- ・ 表記

自SFのW02-ID→R 4 第 2 領域

- ・ 表記の意味

自分のSFのW02パレットIDを、R 4 の第 2 領域に移送する事を意味する。

(31) ベクトル種別がR 2 Cの第 2 規約の表記の説明

- ・ 表記

指定ボタン情報(要件), 下位SFのW04-ID→R 2 C第 2 領域

- ・ 表記の意味

指定ボタン情報(要件)とは、画面のコマンドボタンやファンクションボタンを示す。画面ボタンの種類により下位のSFが決定されるので、其の数分、下位のSFのW04パレットIDを、R 2 Cの第 2 領域に移送する事を意味する。

(32) ベクトル種別がR 2 の第 2 規約の表記の説明

- ・ 表記

自SFのW03-ID→R 2 第 2 領域

- ・ 表記の意味  
自分の S F の W 0 3 パレット I D を, R 2 の第 2 領域に移送する事を意味する。
- (33) ベクトル種別が R 3 R の第 2 規約の表記の説明
- ・ 表記  
自 S F の W 0 4 - I D → R 3 R 第 2 領域
  - ・ 表記の意味  
自分の S F の W 0 4 パレット I D を, R 3 R の第 2 領域に移送する事を意味する。
- (34) ベクトル種別が S 4 で主語が I 2 第 2 領域の第 2 規約の表記の説明
- ・ 表記  
I 2 第 2 領域をクリア
  - ・ 表記の意味  
I 2 第 2 領域を属性クリアする事を意味する。
- (35) ベクトル種別が S 4 で主語が I 2 第 4 領域の第 2 規約の表記の説明
- ・ 表記  
I 2 第 4 領域をクリア
  - ・ 表記の意味  
I 2 第 4 領域を属性クリアする事を意味する。
- (36) ベクトル種別が S 4 で主語が L 2 第 4 領域の第 2 規約の表記の説明
- ・ 表記  
L 2 第 4 領域をクリア
  - ・ 表記の意味  
L 2 第 4 領域を属性クリアする事を意味する。
- (37) ベクトル種別が S 4 で主語が単語／入力アクセスキー／入力処理条件キー／出力アクセスキー／出力処理条件キーの L 3 第 4 領域の第 2 規約の表記の説明
- ・ 表記  
L 3 第 4 領域をクリア
  - ・ 表記の意味  
L 3 第 4 領域を属性クリアする事を意味する。
- (38) ベクトル種別が S 4 で主語が L 4 第 4 領域の第 2 規約の表記の説明
- ・ 表記  
L 4 第 4 領域をクリア
  - ・ 表記の意味  
L 4 第 4 領域を属性クリアする事を意味する。
- (39) ベクトル種別が S 4 で主語が O 4 第 4 領域の第 2 規約の表記の説明
- ・ 表記  
O 4 第 4 領域をクリア
  - ・ 表記の意味  
O 4 第 4 領域を属性クリアする事を意味する。
- (40) ベクトル種別が I 2 の第 3 規約の表記の説明
- ・ 表記  
入力コマンドの実行 S T S の正常判定
  - ・ 表記の意味

入力コマンド命令発報後に得られるステータス値が正常であるかの判定を意味する

。

(41) ベクトル種別がL 2 の第 3 規約の表記の説明

- ・表記  
L 2 第 2 領域＝空
- ・表記の意味  
L 2 の第 2 領域にデータがある事の判定を意味する。

(42) ベクトル種別がL 3 の第 3 規約の表記の説明

- ・表記  
L 4 の実行条件判定, 1 → L 3 第 4 領域
- ・表記の意味  
L 4 の実行条件判定とは, L 4 の第 2 規約に記述する主語の論理性に対応する業務条件式を指す。条件式を判定し, 成立した場合にはL 3 の第 4 領域に 1 を移送する事を意味する。

(43) ベクトル種別がL 3 で属性が等価の第 3 規約の表記の説明

- ・表記  
L 4 の実行条件判定A, 1 → L 3 第 4 領域, L 4 の実行条件判定B, 2 → L 3 第 4 領域
- ・表記の意味  
L 4 の実行条件判定とは, L 4 の第 2 規約に記述する主語の論理性に対応する業務条件式を指す。等価数を 2 と置いた時の例を定義規則図は示しているが, 1 つ目の業務要件を A と定め, 2 つ目の業務要件を B と定めた。A の条件を判定し, 成立した場合にはL 3 の第 4 領域に 1 を移送し, B の条件を判定し, 成立した場合にはL 3 の第 4 領域に 2 を移送する事を意味する。

(44) ベクトル種別がL 3 でK主語の第 3 規約の表記の説明

- ・表記  
派生元の実行条件判定, 1 → L 3 第 4 領域
- ・表記の意味  
派生元の実行条件判定とは, K の派生元となった正規主語の論理性に対応する業務条件式を指す。条件式を判定し, 成立した場合にはL 3 の第 4 領域に 1 を移送する事を意味する。

(45) ベクトル種別がL 3 でK主語で属性が等価の第 3 規約の表記の説明

- ・表記  
派生元の実行条件判定A, 1 → L 3 第 4 領域,  
派生元の実行条件判定B, 2 → L 3 第 4 領域
- ・表記の意味  
派生元の実行条件判定とは, K の派生元となった正規主語の論理性に対応する業務条件式を指す。等価数を 2 と置いた時の例を定義規則図は示しているが, 1 つ目の業務要件を A と定め, 2 つ目の業務要件を B と定めた。A の条件を判定し, 成立した場合にはL 3 の第 4 領域に 1 を移送し, B の条件を判定し, 成立した場合にはL 3 の第 4 領域に 2 を移送する事を意味する。

(46) ベクトル種別がL 3 で入力アクセスキー主語の第 3 規約の表記の説明

- ・表記  
入力キー成立条件の判定, 1 → L 3 第 4 領域

- ・表記の意味

入力キー成立条件の判定とは、I 2 の入力キーが成立する条件を指す。成立条件を判定し、成立した場合には入力アクセスキーの L 3 の第 4 領域に 1 を移送する事を意味する。

(47) ベクトル種別が L 3 で入力処理条件キー主語の第 3 規約の表記の説明

- ・表記

入力処理条件の判定, 1 → L 3 第 4 領域

- ・表記の意味

入力処理条件の判定とは、I 2 の処理条件を指す。処理条件を判定し、成立した場合には入力処理条件キーの L 3 の第 4 領域に 1 を移送する事を意味する。

(48) ベクトル種別が L 3 で出力アクセスキー主語の第 3 規約の表記の説明

- ・表記

出力キー成立条件の判定, 1 → L 3 第 4 領域

- ・表記の意味

出力キー成立条件の判定とは、O 4 の出力キーが成立する条件を指す。成立条件を判定し、成立した場合には出力アクセスキーの L 3 の第 4 領域に 1 を移送する事を意味する。

(49) ベクトル種別が L 3 で出力処理条件キー主語の第 3 規約の表記の説明

- ・表記

出力処理条件の判定, 1 → L 3 第 4 領域

- ・表記の意味

出力処理条件の判定とは、O 4 の処理条件を指す。処理条件を判定し、成立した場合には出力処理条件キーの L 3 の第 4 領域に 1 を移送する事を意味する。

(50) ベクトル種別が L 4 の第 3 規約の表記の説明

- ・表記

L 4 第 2 領域 = 空

- ・表記の意味

L 4 の第 2 領域にデータがある事の判定を意味する。

(51) ベクトル種別が O 4 の第 3 規約の表記の説明

- ・表記

出力コマンドの実行 S T S の正常判定

- ・表記の意味

出力コマンド命令発報後に得られるステータス値が正常である事の判定を意味する。

。

(52) ベクトル種別が R 4 の第 3 規約の表記の説明

- ・表記

R 4 第 2 領域 = 空

- ・表記の意味

R 4 の第 2 領域にデータがある事の判定を意味する。

(53) ベクトル種別が R 2 C の第 3 規約の表記の説明

- ・表記

R 2 C 第 2 領域 = 空

- ・表記の意味

R 2 C の第 2 領域にデータがある事の判定を意味する。

(54) ベクトル種別が R 2 の第 3 規約の表記の説明

- ・表記  
R 2 第 2 領域 = 空
- ・表記の意味  
R 2 の第 2 領域にデータがある事の判定を意味する。

(55) ベクトル種別が R 3 R の第 3 規約の表記の説明

- ・表記  
R 3 R 第 2 領域 = 空
- ・表記の意味  
R 3 R の第 2 領域にデータがある事の判定を意味する。

(56) ベクトル種別が S 4 の第 3 規約の表記の説明

- ・表記  
NOP
- ・表記の意味  
ノーオペレーションを意味する。

(57) ベクトル種別が I 2 の第 4 規約の表記の説明

- ・表記  
入力コマンドの結果 → S T S の保持領域, I 2 今回状態変化フラグに 1 を加算
- ・表記の意味  
入力コマンドの結果とは, 入力コマンド命令発報後に得られる正常ステータスを指す。正常ステータス値の種別分ステータス保持領域を設定し, ステータスに対応した保持領域に識別子を移送すると共に, 自分の S F の I 2 今回状態変化フラグに 1 を加算する事を意味する。

(58) ベクトル種別が L 2 の第 4 規約の表記の説明

- ・表記  
L 2 第 2 領域 → L 2 第 4 領域, L 2 今回状態変化フラグに 1 を加算
- ・表記の意味  
L 2 の第 2 領域のデータを第 4 領域に移送すると共に, 自分の S F の L 2 今回状態変化フラグに 1 を加算する事を意味する。

(59) ベクトル種別が L 3 の第 4 規約の表記の説明

- ・表記  
L 3 今回状態変化フラグに 1 を加算
- ・表記の意味  
自分の S F の L 3 今回状態変化フラグに 1 を加算する事を意味する。

(60) ベクトル種別が L 4 の第 4 規約の表記の説明

- ・表記  
L 4 第 2 領域 → L 4 第 4 領域, L 4 今回状態変化フラグに 1 を加算
- ・表記の意味  
L 4 の第 2 領域のデータを第 4 領域に移送すると共に, 自分の S F の L 4 今回状態変化フラグに 1 を加算する事を意味する。

(61) ベクトル種別が L 4 で K 主語の第 4 規約の表記の説明

- ・表記



L 4 第 2 領域→L 4 第 4 領域, L 4 今回状態変化フラグに 1 を加算, L 4 第 4 領域  
→L 2 第 4 領域

・表記の意味

L 4 の第 2 領域のデータを第 4 領域に移送すると共に, 自分の S F の L 4 今回状態  
変化フラグに 1 を加算し, 第 4 領域のデータを L 2 の第 4 領域にも移送する事を意味する  
。

(62) ベクトル種別が L 4 で属性が境界の第 4 規約の表記の説明

・表記

L 4 第 2 領域→L 4 第 4 領域, L 4 今回状態変化フラグに 1 を加算, L 4 第 4 領域  
→S F 1 L 4 第 4 領域

・表記の意味

L 4 の第 2 領域のデータを第 4 領域に移送すると共に, 自分の S F の L 4 今回状態  
変化フラグに 1 を加算し, 第 4 領域のデータを隣々上位 (図は S F 1 で示した例) の L 4  
の第 4 領域にも移送する事を意味する。

(63) ベクトル種別が O 4 の第 4 規約の表記の説明

・表記

出力コマンドの結果→S T S の保持領域, O 4 今回状態変化フラグに 1 を加算

・表記の意味

出力コマンドの結果とは, 出力コマンド命令発報後に得られる正常ステータスを指  
す。正常ステータス値の種別分ステータス保持領域を設定し, ステータスに対応した保持  
領域に識別子を移送すると共に, 自分の S F の O 4 今回状態変化フラグに 1 を加算する事  
を意味する。

(64) ベクトル種別が R 4 の第 4 規約の表記の説明

・表記

R 4 第 2 領域→R 4 第 4 領域

・表記の意味

R 4 の第 2 領域のデータを, 第 4 領域に移送する事を意味する。

(65) ベクトル種別が R 2 C の第 4 規約の表記の説明

・表記

R 2 C 第 2 領域→R 2 C 第 4 領域

・表記の意味

R 2 C の第 2 領域のデータを, 第 4 領域に移送する事を意味する。

(66) ベクトル種別が R 2 の第 4 規約の表記の説明

・表記

R 2 第 2 領域→R 2 第 4 領域

・表記の意味

R 2 の第 2 領域のデータを, 第 4 領域に移送する事を意味する。

(67) ベクトル種別が R 3 R の第 4 規約の表記の説明

・表記

R 3 R 第 2 領域→R 3 R 第 4 領域

・表記の意味

R 3 R の第 2 領域のデータを, 第 4 領域に移送する事を意味する。

## (68) ベクトル種別が S 4 の第 4 規約の表記の説明

- ・表記  
NOP
- ・表記の意味  
ノーオペレーションを意味する。

## (69) ベクトル種別が I 2 の第 5 規約の表記の説明

- ・表記  
入力コマンドの実行 S T S の異常判定
- ・表記の意味  
入力コマンド命令発報後に得られるステータス値が異常であるかの判定を意味する

。

## (70) ベクトル種別が L 2 の第 5 規約の表記の説明

- ・表記  
L 2 前回状態変化フラグ = L 2 前々回状態変化フラグ

- ・表記の意味  
自分の S F の L 2 前回状態変化フラグと L 2 前々回状態変化フラグを比較し、同じであるかを判定する事を意味する。

## (71) ベクトル種別が L 3 の第 5 規約の表記の説明

- ・表記  
L 3 前回状態変化フラグ = L 3 前々回状態変化フラグ

- ・表記の意味  
自分の S F の L 3 前回状態変化フラグと L 3 前々回状態変化フラグを比較し、同じであるかを判定する事を意味する。

## (72) ベクトル種別が L 4 の第 5 規約の表記の説明

- ・表記  
L 4 前回状態変化フラグ = L 4 前々回状態変化フラグ

- ・表記の意味  
自分の S F の L 4 前回状態変化フラグと L 4 前々回状態変化フラグを比較し、同じであるかを判定する事を意味する。

## (73) ベクトル種別が O 4 の第 5 規約の表記の説明

- ・表記  
出力コマンドの実行 S T S の異常判定

- ・表記の意味  
出力コマンド命令発報後に得られるステータス値が異常であるかの判定を意味する

。

## (74) ベクトル種別が R 4 の第 5 規約の表記の説明

- ・表記  
R 4 第 2 領域 ≠ 空

- ・表記の意味  
R 4 の第 2 領域にデータがない事の判定を意味する。

## (75) ベクトル種別が R 2 C の第 5 規約の表記の説明

- ・表記  
R 2 C 第 2 領域 ≠ 空

- ・ 表記の意味  
R 2 C の第 2 領域にデータがない事の判定を意味する。
- (76) ベクトル種別が R 2 の第 5 規約の表記の説明
- ・ 表記  
R 2 第 2 領域 ≠ 空
  - ・ 表記の意味  
R 2 の第 2 領域にデータがない事の判定を意味する。
- (77) ベクトル種別が R 3 R の第 5 規約の表記の説明
- ・ 表記  
R 3 R 第 2 領域 ≠ 空
  - ・ 表記の意味  
R 3 R の第 2 領域にデータがない事の判定を意味する。
- (78) ベクトル種別が S 4 の第 5 規約の表記の説明
- ・ 表記  
NOP
  - ・ 表記の意味  
ノーオペレーションを意味する。
- (79) ベクトル種別が S 4 以外の第 6 規約の表記の説明
- ・ 表記  
ON → 不成立フラグ
  - ・ 表記の意味  
不成立フラグに ON を設定する事を意味する。
- (80) ベクトル種別が S 4 の第 6 規約の表記の説明
- ・ 表記  
NOP
  - ・ 表記の意味  
ノーオペレーションを意味する。
- (81) ベクトル種別が S 4 の第 7 規約の表記の説明
- ・ 表記  
NOP
  - ・ 表記の意味  
ノーオペレーションを意味する。
- (82) ベクトル種別が S 4 以外の全ての第 7 規約の表記の説明
- ・ 表記  
ON → 再起フラグ
  - ・ 表記の意味  
再起フラグに ON を設定する事を意味する。

定義規則を以下に説明する。

- (01) 定義規則 1：主語が正規で属性が入力のベクトル  
正規単語で其の属性が入力の主語は定義規則 1 で規約される L 2 の型のベクトルとなる。定義規則は定義規則図 1 参照。

正規単語は入出力論理体に属す単語、換言すれば開発案件に属す単語の事である。其の情報は L y e e B E L T 上で規約される。定義規則図 1 の左側のフローチャートは P S の構造を示し、右側の L 2 規約とあるのはこの主語のベクトルが P S の構造規則で述べた L 2 の型を基に規約されるからである。

以下に、この場合の定義規則を総括的に述べる。

【0012】

1 第1規約の命令はこのベクトルが最初に実行される命令である。其の命令は「自分のベクトルの第4領域のデータの有無を問う」判定命令となる。第4領域とはこのベクトルの第4規約の命令で其の状態が決定される領域である。第三章の P S (図14乃至16)の視点から第4領域の意義を述べれば、第4領域は主語である自分の子孫が捜し求められた事を代理し、且つ主語である自分の存在の所在を状態として表す為の場である。其れ故、第4領域の識別子は主語の識別子に対応して決められなければならない。其して、其の領域の属性は主語の記憶装置上の属性(文字型、桁、量)に対応して決められる。この事は P S の導出の論理を成立させる必然的な律性である。

定義規則の規約にしばしば現れる「空」の意義をここで説明する。即ち、本研究では既述の様にソフトは意識の存在として定義され、其れを捉える事がソフトであると定義されている。他方、其の意識の存在をわれわれが捉える事は不可能である。即ち、意識の存在から選ばれる全ての存在が捉えられれば、其れは意識の存在を捉えるに等しい。しかし、われわれが記憶出来る状態の数は其れら存在の数の部分に過ぎない事になるからである。其れ故、本研究では其れら存在ひとつひとつが全存在と同等であると見なす事が出来るならば、其の全存在を捉えなくてもわれわれは記憶出来るだけの存在で其の全存在を捉えるに等しい関係を成立させる事が出来る。この為に I D S の定義(第五章:16参照、図4)で定義される空の概念が利用される。即ち、空の概念は本研究の基本概念で言う履歴の中に同様に其の代表空間が属す存在が成立すれば、其の存在の定義から、其の存在の履歴は全体の性質を有する事になる。

ここで、第4領域を自分の履歴と考える事が出来るので、其の領域の中に自分のデータが存在すれば其の状態は空に外ならない。其れ故、第1規約の命令は第4領域が空かと問うに等しい、換言すれば、この問いは自分が全存在を代表しているかどうかとの自問に外ならない。第4領域が空ならば、このベクトルの役割は完了する。其の場合、このベクトルはこの第1規約の命令の以降に置かれている命令を実行する必要はない。因みに、この命令で其のデータの意味的内容を問う必要はない。単に其の領域のデータの有無を問うだけで十分である。

2 第2規約は「自分のベクトルの第4領域の状態を空にする」前段の作用を行うデータ移動命令を規約する。

【0013】

この命令により主語の識別子と一義的に対応する領域の状態が無条件でこのベクトルの第2領域に複写される。主語の識別子と一義的に対応する領域とは、この場合、入力論理体に属す領域の事である。入力論理体とは、例えば入力コマンドで入力される D B の論理レコードが記憶装置上で成立させる領域である。第2領域の属性は第4領域の属性と同じに決められなければならない。但し、其の領域は第4領域の仮の領域を意味するので、第4領域とは識別子を変えて定義される。入力となる論理体に属すデータ項目名がそれぞれ定義規則1で規約されるベクトルの主語である。

3 第3規約は「自分のベクトルの第2領域のデータの有無を問う」判定命令を規約する。

【0014】

換言すれば、第1規約の場合と同じで、第2領域の状態が空かどうかを判定する事である。第2領域が空であれば、第4規約の命令が実行される。第2領域が空でなければ、第5規約の命令が実行される。

4 第4規約は第2領域が空の場合に「自分のベクトルの第2領域のデータを第4領域に複写する」データ複写命令を規約する。

【0015】

この命令は「自分のベクトルの第4領域の状態を空にする」後段の作用である。この命令の実行により、このベクトルの第4領域は空となり、其の事に於いてこのベクトルの役割は完了する。

5 第5規約はL2の第5規約の命令では全L2の第4領域の集合の前回と前々回の空の成立数の合算値（カウンタ）が比較され其の相違の有無が判定される。

【0016】

この比較に相違があれば、自分の第4領域も今回は空を成立させる事が出来なかったが、ベクトル、SF並びにPRDの構造関係から次回の実行の機会には空になれる可能性があると判断出来る。其れを期してこの場合は第7規約の命令が実行される。

空が成立すれば、L2の第4規約の命令で其の事を示す値1がカウンタに加算される。このカウンタは、3種のパレットごとに今回、前回、其して前々回の3種が設けられる。状態に変化がなければ、既に自分には実行経験があるにもかかわらず、其の状態が継続する事であるから、今後も自分の第4領域は空になる事がないと判断出来る。この場合は第6規約の命令が実行される。以上の事はL3、L4の場合も同じである。

6 第6規約は主語の祖先が本来的に存在していない場合はベクトルに再起動停止の機構が成立し、その基でベクトルは再起動を停止し主語の祖先の探索を停止する。

【0017】

必然的な再起動停止の機構は第5規約で規約される命令と第6規約で規約される命令で構築される。「このベクトルの再起動を停止させるフラグをセットする」為のセット命令を定義する。

7 第7規約は主語の祖先が同じENWの中にいない事を意味する。換言すれば、本来的に存在しているが、ある同期状態においては探索出来なかった可能性があると判断される場合、再度、同期状態を希求する為の措置である。「このベクトルの再起動を希求させるフラグをセットする」為のセット命令を定義する。

(02) 定義規則2：主語が正規で属性が入力、配列のベクトル

正規単語で其の属性が入力、且つ配列をなす主語は定義規則2で規約されるL2の型のベクトルとなる。定義規則は定義規則図2参照。

配列とは主語と同じで第2規約も同じであるが、其の第4領域が別個に定義される事を表明する概念（属性）である。其れ故、主語Aの配列をm行、n列とすれば、主語Aのベクトルを構成する全ての領域はそれぞれmとnの積の数だけ規約される。これら領域は主語との関係で付される識別子にmとnの対で定義される補助的な識別子が付されてそれぞれ規約される。

この事から、配列を扱うこのベクトルはPSで規約される構造の外に、配列分の繰り返し操作を自律的に行う仕掛（LOOP）が必要である。これが他のベクトルの構造と異なる所である。LOOPを行う配列のmn情報、並びにLOOPがmnのどちらから行われるかは、LyeeBELTで捉えられる開発案件に属す情報である。LOOP以外の

ベクトルの規約の仕方は定義規則 1 と同じである。

(03) 定義規則 3：主語が正規で属性が出力のベクトル

正規単語で其の属性が出力をなす主語は、定義規則 3 で規約される L 3, L 4 の型のベクトルとなる。定義規則は定義規則図 3 参照。

主語の属性が出力の場合は其の主語に対し必ず L 3, L 4 が対で作られなければならない。これらベクトルの規約の仕方は定義規則図 3 で示される。L 3 の L 4 に対する役割は自分の L 4 に実行許可を与える事である。L 3 の第 4 領域には L 3 自身により其の為の情報がセットされる。L 4 は自分の L 3 の第 4 領域と自分の第 4 領域の状態を自分の第 1 規約の命令で判定する。定義規則図 3 で示されている L 4 の第 1 規約から L 3 の第 4 規約の位置（第 4 領域）に向かう矢線はこの関係を示す為のものである。因みに、定義規則図に表されているこの形式の矢線を本研究では参照線と呼ぶ。

第 1 規約の命令の判定結果で実行が必要になれば第 2 規約の命令を実行し、必要がなければこのベクトルの実行は終了する。どの型のベクトルもそれぞれ独立して作られる。しかし、ベクトルが S F 又は P R D の要素として実行される際、参照線で示される様に領域の参照関係が生じ論理的従属関係が成立する。即ち、この参照線が既述の相補作用に外ならない。

(04) 定義規則 4：主語が正規で属性が出力、等価のベクトル

正規単語で其の属性が出力且つ等価をなす主語は、定義規則 4 で規約される L 3, L 4 の型のベクトルとなる。定義規則は定義規則図 4 参照。

この定義規則は定義規則 3 に以下に述べる規則が追加されるものである。主語の識別子は同じで等価の属性が付加されて規約される主語の L 4 のベクトルは、其の付加される等価の数分作られる。しかし、等価数と関係なく L 3 は 1 個である。定義規則図 4 では等価数が 2 個の場合が示されている。等価は主語が同じでも第 2 規約の命令の内容が異なる。しかし、其の第 4 領域が共通で 1 個で済む事を表明する概念（属性）である。其それぞれの L 4 は其の第 1 規約の命令で共通の L 3 の第 4 領域を参照し、自分が実行されるべきかどうかを判定する。この場合の実行されるべき L 4 の中のひとつである。

(05) 定義規則 5：主語が正規で属性が出力、配列のベクトル

正規単語で其の属性が出力且つ配列をなす主語は、定義規則 5 で規約される L 3, L 4 の型のベクトルとなる。定義規則は定義規則図 5 参照。

この定義規則は定義規則 3 に以下に述べる規則が追加されるものである。この場合の L 3, L 4 はベクトルはそれぞれ 1 個であるが L 3, L 4 を構成するベクトルの領域の数は配列の数だけ規約される。其して、この L 4 のベクトルの第 1 規約の命令が参照する L 3 のベクトルの第 4 領域は配列で対となる L 3 の第 4 領域である。

(06) 定義規則 6：主語が正規で属性が出力、等価、配列のベクトル

正規単語で其の属性が出力、等価、且つ配列をなす主語は、定義規則 4, 5 の合成された定義規則で規約される L 3, L 4 の型のベクトルとなる。定義規則は定義規則図 6 参照。

(07) 定義規則 7：主語が正規で属性が出力、境界のベクトル

正規単語で其の属性が出力、且つ境界をなす主語は定義規則 7 で規約される L 3, L 4 の型のベクトルとなる。定義規則は定義規則図 7 参照。

この規則は定義規則 3 に以下に述べる規則が追加されるものである。境界の概念は 4.2 (5) 又は表 3 で示されている。L 4 のベクトルの第 2 規約の始点が自分が属す S F の W 0 4, W 0 2, 其して自分が属す S F の隣下位の S F の W 0 4 に存在しなければ, 其の始点を主語 (端点) とする L 4 は隣隣下位の S F の W 0 4 に属すベクトルである。この隣隣下位の S F の W 0 4 に属す L 4 のベクトルの主語の属性が境界である。この場合, 其の L 4 のベクトルの第 4 領域の記憶上の属性と同じ領域が隣隣上位の S F の

W 0 4 に設けられる。其して, 境界となる L 4 のベクトルの第 4 規約には定義規則 3 の第 4 規約の命令の外に, 自分の第 4 領域の状態を其の領域に移送する命令が規約される。この措置により, 隣隣上位の S F の W 0 4 の L 4 のベクトルは其の第 2 規約の命令の始点を正常に獲得する事が出来る。

(08) 定義規則 8 : 主語が正規で属性が出力, 等価, 境界のベクトル

正規単語で其の属性が出力, 等価, 且つ境界をなす主語は, 定義規則 4, 7 の合成された定義規則で規約される L 3, L 4 の型のベクトルとなる。定義規則は定義規則図 8 参照。

(09) 定義規則 9 : 主語が正規で属性が出力, 配列, 境界のベクトル

正規単語で其の属性が出力, 配列, 且つ境界をなす主語は, 定義規則 5, 7 の合成された定義規則で規約される L 3, L 4 の型のベクトルとなる。定義規則は定義規則図 9 参照。

(10) 定義規則 10 : 主語が正規で属性が出力, 等価, 配列, 境界のベクトル

正規単語で其の属性が出力, 等価, 配列, 且つ境界をなす主語は, 定義規則 4, 5, 7 が合成された定義規則で規約される L 3, L 4 の型のベクトルとなる。定義規則は定義規則図 10 参照。

(11) 定義規則 11 : 主語が K のベクトル

集計の為の操作領域の名称を総称して K と呼ぶ。K はベクトルとして其れ自体の論理性と領域を有するが正規単語に付属して存在するものである。K が主語となる場合は定義規則 11 で規約される L 2, L 3, L 4 の型のベクトルとなる。L 3 と L 4 の間の参照線の関係は定義規則 3 の場合と同じである。L 2 はこの L 4 の第 4 領域と同じ領域が設けられる。L 4 の第 4 規約の命令は定義規則 3 の規約の外に設けられるこの L 2 の領域に自分の第 4 領域の状態を移送する命令が加えられる。定義規則は定義規則図 11 参照。

(12) 定義規則 12 : 主語が K で属性が等価のベクトル

属性が等価の K が主語となる場合は, 定義規則 11, 4 が合成された本定義規則で規約される L 2, L 3, L 4 の型のベクトルとなる。この定義規則は定義規則図 12 参照。

(13) 定義規則 13 : 主語が K で属性が配列のベクトル

属性が配列の K が主語となる場合は, 定義規則 11, 5 が合成された本定義規則で規約される L 2, L 3, L 4 の型のベクトルとなる。定義規則は定義規則図 13 参照。

(14) 定義規則 14 : 主語が K で属性が等価, 配列のベクトル

属性が配列, 等価の K が主語となる場合は定義規則 12, 13 が合成された本定義規則で規約される L 2, L 3, L 4 の型のベクトルとなる。定義規則は定義規則図 14 参照。

(15) 定義規則 15 : 主語が M で属性が出力のベクトル

見失われていた正規単語を正規単語として扱わない場合、其の単語をMと呼ぶ。Mは其れ 自体の内部論理性と領域を有する。属性が出力のMが主語となる場合は、定義規則 15 で規約される L 3, L 4 の型のベクトルとなる。定義規則は定義規則図 15 参照, この定義規則は定義規則 3 に準拠する。

(16) 定義規則 16 : 主語がMで属性が出力, 等価のベクトル  
属性が出力, 且つ等価のMが主語となる場合は, 定義規則 16 で規約される L 3, L 4 の型のベクトルとなる。定義規則は定義規則図 16 参照, この定義規則は定義規則 4 に準拠する。

(17) 定義規則 17 : 主語がMで属性が出力, 配列のベクトル  
属性が出力, 且つ配列のMが主語となる場合は, 定義規則 17 で規約される L 3, L 4 の型のベクトルとなる。定義規則は定義規則図 17 参照, この定義規則は定義規則 5 に準拠する。

(18) 定義規則 18 : 主語がMで属性が出力, 等価, 配列のベクトル  
属性が出力, 等価, 配列のMが主語となる場合は, 定義規則 18 で規約される L 3, L 4 の型のベクトルとなる。定義規則は定義規則図 18 参照, この定義規則は定義規則 6 に準拠する。

(19) 定義規則 19 : 主語が論理体で属性が入力のベクトル  
属性が入力で論理体が主語となる場合は, 定義規則 19 で規約される I 2 の型のベクトルとなる。定義規則は定義規則図 19, 20, 21 参照。I 2 の領域の関係は図 23, I 2 の作成単位は図 24 に示す。

(20) 定義規則 20 : 主語がアクセスキーで属性が出力のベクトル  
属性が出力で入力アクセスキーが主語となる場合は, 定義規則 20 で規約される入力アクセスキーの型の L 3 のベクトルとなる。L 4 のベクトルが存在しないのは, 定義規則 19 の I 2 が其の役割を代行する為である。定義規則は定義規則図 19, 20, 21 参照。

(21) 定義規則 21 : 主語が処理条件キーで属性が出力のベクトル  
属性が出力で入力処理条件キーが主語となる場合は, 定義規則 21 で規約される入力処理条件キーの型の L 3 のベクトルとなる。

**【0018】**

L 4 のベクトルが存在しないのは, 定義規則 19 の I 2 が其の役割を代行する為である。定義規則は定義規則図 19, 20, 21 参照。

(22) 定義規則 22 : 主語が論理体で属性が出力のベクトル  
属性が出力で論理体が主語となる場合は, 定義規則 22 で規約される O 4 の型のベクトルとなる。定義規則は定義規則図 22, 23, 24 参照。O 4 の領域の関係は図 25, O 4 の作成単位は図 26 に示す。

(23) 定義規則 23 : 主語がアクセスキーで属性が出力のベクトル  
属性が出力で出力アクセスキーが主語となる場合は, 定義規則 23 で規約される出力アクセスキーの型の L 3 のベクトルとなる。L 4 のベクトルが存在しないのは, 定義規則 22 の O 4 が其の役割を代行する為である。定義規則は定義規則図 22, 23, 24 参照。

(24) 定義規則 24 : 主語が処理条件キーで属性が出力のベクトル  
属性が出力で出力処理条件キーが主語となる場合は, 定義規則 24 で規約される出



力処理条件キーの型の L 3 のベクトルとなる。L 4 のベクトルが存在しないのは、定義規則 22 の O 4 が其の役割を代行する為である。定義規則は定義規則図 22, 23, 24 参照。

(25) 定義規則 25：主語がパレット W 0 4 で R 4 のベクトル

パレット種別 W 0 4 が主語となる場合は、定義規則 25 で規約される R 4 の型の L 4 のベクトルとなる。定義規則は定義規則図 25 参照。

(26) 定義規則 26：主語がパレット W 0 2 で R 2 C のベクトル

パレット種別 W 0 2 が主語となる場合で操作要件が存在する場合には、R 4 と異なる定義規則 26 で規約される R 2 C の型の L 4 のベクトルとなる。定義規則は定義規則図 26 参照。

(27) 定義規則 27：主語がパレット W 0 2 で R 2 のベクトル

パレット種別 W 0 2 が主語となる場合は、定義規則 27 で規約される R 2 の型の L 4 のベクトルとなる。定義規則は定義規則図 27 参照。

(28) 定義規則 28：主語がパレット W 0 3 で R 3 R のベクトル

パレット種別 W 0 3 が主語となる場合は、定義規則 28 で規約される R 3 R の型の L 4 のベクトルとなる。定義規則は定義規則図 28 参照。

(29) 定義規則 29：主語が I 2 第 2 領域の S 4 のベクトル

入力作用要素 (I 2) の第 2 領域が主語となる場合は、定義規則 29 で規約される S 4 の型のベクトルとなる。これは、入力作用要素 (I 2) の第 2 領域の実行時の非空処理 (クリア) を行う為のベクトルである。

(30) 定義規則 30：主語が I 2 第 4 領域の S 4 のベクトル

入力作用要素 (I 2) の第 4 領域が主語となる場合は、定義規則 30 で規約される S 4 の型のベクトルとなる。これは、入力作用要素 (I 2) の第 4 領域の実行時の非空処理 (クリア) を行う為のベクトルである。

(31) 定義規則 31：主語が入力アクセスキーの L 3 第 4 領域の S 4 のベクトル

入力アクセスキーの L 3 第 4 領域が主語となる場合は、定義規則 31 で規約される S 4 の型のベクトルとなる。これは、入力アクセスキーの L 3 第 4 領域の実行時の非空処理 (クリア) を行う為のベクトルである。

(32) 定義規則 32：主語が入力処理条件キーの L 3 第 4 領域の S 4 のベクトル

入力処理条件キーの L 3 第 4 領域が主語となる場合は、定義規則 32 で規約される S 4 の型のベクトルとなる。これは、入力処理条件キーの L 3 第 4 領域の実行時の非空処理 (クリア) を行う為のベクトルである。

(33) 定義規則 33：主語が O 4 第 4 領域の S 4 のベクトル

出力作用要素 (O 4) の第 4 領域が主語となる場合は、定義規則 33 で規約される S 4 の型のベクトルとなる。これは、出力作用要素 (O 4) の第 4 領域の実行時の非空処理 (クリア) を行う為のベクトルである。

(34) 定義規則 34：主語が出力アクセスキーの L 3 第 4 領域の S 4 のベクトル

出力アクセスキーの L 3 第 4 領域が主語となる場合は、定義規則 34 で規約される S 4 の型のベクトルとなる。これは、出力アクセスキーの L 3 第 4 領域の実行時の非空処理 (クリア) を行う為のベクトルである。

(35) 定義規則 35: 主語が出力処理条件キーの L3 第 4 領域の S4 のベクトル  
出力処理条件キーの L3 第 4 領域が主語となる場合は、定義規則 35 で規約される S4 の型のベクトルとなる。これは、出力処理条件キーの L3 第 4 領域の実行時の非空処理 (クリア) を行う為のベクトルである。

(36) 定義規則 36: 主語が L4 第 4 領域の S4 のベクトル  
L4 第 4 領域が主語となる場合は、定義規則 36 で規約される S4 の型のベクトルとなる。これは、L4 第 4 領域の実行時の非空処理 (クリア) を行う為のベクトルである。定義規則 37 から 47 についても定義規則 36 と同じである。

(37) 定義規則 48: 主語が L2 第 4 領域の S4 のベクトル  
L2 第 4 領域が主語となる場合は、定義規則 48 で規約される S4 の型のベクトルとなる。これは、L2 第 4 領域の実行時の非空処理 (クリア) を行う為のベクトルである。定義規則 49 から 53 についても定義規則 48 と同じである。

(38) 定義規則 54: 主語が L3 第 4 領域の S4 のベクトル  
L3 第 4 領域が主語となる場合は、定義規則 54 で規約される S4 の型のベクトルとなる。これは、L3 第 4 領域の実行時の非空処理 (クリア) を行う為のベクトルである。定義規則 55 から 69 についても定義規則 54 と同じである。

#### 4. 12 プログラムの公式のスキーム

本研究で言うプログラムの公式とは、SF または PRD を決定するアルゴリズムの事である。其のアルゴリズムにより、ベクトル、パレット関数、パレット連鎖関数が定義される。其のアルゴリズムは、図 267 の様に整理出来る。

ベクトルの定義規則は、記述の通り決定的である。其れを用いてプログラム言語に応じてベクトルの型を定義する事が出来る。同じ様に、パレット関数の定義規則、パレット連鎖関数の定義規則も定義出来る。其して、これらの型に L y e e B E L T 情報を機械的 (自動的) に対応付ければ、本研究で言うプログラム、即ち、SF、並びに PRD を定義する事が出来る。

本論文では、プログラム言語が VB の場合の型を添付する。定義されるベクトルの全型の一覧表を、表 10 (10-1 ~ 10-3) に示す。同様に、L y e e B E L T の例を表 11、ベクトルの型の例を表 12 (12-1, 12-2)、ベクトルの型から作り出されるベクトルを表 13 (13-1, 13-2)、パレット関数の型の例を表 14 (14-1 ~ 14-8)、其して、パレット連鎖関数の型の例を表 15 (15-1 ~ 15-15) に示す。

付録は、表 10 に基づく全ベクトルの型、パレット関数の型、パレット連鎖関数の型である。これは、プログラムの公式 (第四章 4. 6) の成立性を実証する為に添付されている。

### 第五章 IDS に成立する定義の解説

存在の基本概念 (第二章 2. 1) を仮説概念 (第二章 2. 2) を用いてモデル化し、其のモデル を IDS (IDeal Space) と称し、其処に成立する規則を IDS の定義と呼ぶ。IDS の模式図は、図 1 で示される。以下に IDS の定義として成立する定義を解説する。これらの定義は、存在

の振舞い、PS、TDM等を定義する為に使われる。

#### 01：境界時間速度 ( $V_B$ )

不可知空間（第二章参照）の外に位相される時間速度は、集合の要素となる。IDSとは、この集合の事である。IDSには、同じ時間速度が要素として属す事はない。IDSに属す時間速度が3個以上になれば、其の値の大きな方を左に置き順次右へ並べる順序列が成立する。其の順列の midpoint に位置する時間速度を境界時間速度と呼ぶ。境界時間速度はこの時点の時間速度の集合の代表となる。其れを  $V_B$  で表す。新たな時間速度が位相されれば、其の都度、改めて境界時間速度が選ばれ、IDSが再構築される。境界時間速度は、時間速度の順序列で左側に属す時間速度の個数が多くなる様に選ばれる。

#### 02：時間速度 ( $V_i$ ) の位置活力

時間速度  $V_i$  の位置活力は  $|V_i - V_B|$  である。境界時間速度が選ばれるとそれぞれの時間速度の位置活力が決定される。

#### 03： $V_B$ の位置活力

$V_B$  の位置活力は、ゼロではなく  $\epsilon$  に置き換えられる。 $\epsilon$  は、最小の時間速度の  $1/2$  の値で特異数となる。図1参照。

#### 04： $V_i$ の空間観念

$V_i$  の空間観念は  $1/|V_i - V_B|$  で、体積空間を意味する。成立条件は、 $1/|V_i - V_B|$  と同じ値の時間速度がIDSに属している事である。本研究では、2つの存在AとBに於いて、Aの空間観念がBの空間観念よりも大きければ、其の時AはBを内包し、BはAを外延する。換言すれば、Bは時間的に過去の存在、Aは時間的にBよりも未来の存在として位置付けられる。図1、図2参照。

#### 05：境界時間速度の空間観念

境界時間速度の空間観念は  $1/\epsilon$  である。 $1/\epsilon$  は定数となる。其れ故、境界時間速度の空間観念は恒常的に成立するものである。この値は臨界自然数  $\Phi$  と一致するものとし、空間観念としては最大である。其れ故、境界原子の空間観念がIDSを代表する空間を表す。臨界自然数は特異数である。図1参照。

#### 06：存在線

空間観念が成立する時間速度の集合の事である。

#### 07：論理原子

境界時間速度を含む境界時間速度よりも大きな時間速度を意識原子、其れ以外の時間速度を認識原子、総称して論理原子と呼ぶ。特に、境界時間速度を境界原子と呼ぶ。論理原子は、時間速度、位置活力、空間観念の組として定義される。図2参照。

#### 08：不可知空間の着座座標

IDSが構築され、存在線が成立すると不可知空間は境界原子の時間速度を座標として存在線に属す。図1参照。

#### 09：論理原子の着座座標

認識原子は、自分の時間速度の値を座標として存在線に属す。境界原子は、自分の時間速度プラス  $\epsilon$  の値を座標として存在線に属す。意識原子は、境界原子と同じ座標に収斂して存在線に属す。図1参照。

10： $\lambda$  集合

$\lambda$  集合とは、論理要素の数が3個以上の奇数からなる集合で、かつ、同じ $\lambda$ 集合では同じ論理要素が重複して属する事がない集合の事である。この集合は存在線上で成立する。意識原子を要素とする $\lambda$ 集合を意識 $\lambda$ 集合、認識原子を要素とする $\lambda$ 集合を認識 $\lambda$ 集合と呼ぶ。

## 11：占有空間

論理原子の集合に属す論理原子の空間観念の総和を占有空間と呼ぶ。占有空間は、体積的空間を意味する。図3参照。

12：意識 $\lambda$ 集合の順列

全意識 $\lambda$ 集合を要素とする順列のひとつを意識 $\lambda$ 集合の順列と言う。  
意識 $\lambda$ 集合の順列は、IDSが再構築される都度、決定される。

13：認識 $\lambda$ 集合の順列

全認識 $\lambda$ 集合を要素とする重複順列のひとつを認識 $\lambda$ 集合の順列と言う。認識 $\lambda$ 集合の順列は、IDSが再構築される都度、決定される。

## 14：空

$\lambda$ 集合の占有空間に可能的に内側で近似する空間観念を有する其の $\lambda$ 集合に属す論理原子を選ぶ作用を空と呼ぶ。図4参照。

## 15：意識連鎖

意識 $\lambda$ 集合の順列が定義されると、其の順列の意識 $\lambda$ 集合の順番に従ってそれぞれ空を成立させる意識原子が可能な限り定義される。この空を成立させる意識原子と、意識 $\lambda$ 集合の関係を意識連鎖と呼ぶ。意識連鎖となる意識 $\lambda$ 集合は履歴、空を成立させる意識原子の空間観念を其の意識連鎖の代表空間と呼ぶ。代表空間となる意識原子はIDSの作用で複写され、IDSの空間観念の外に位相される。意識連鎖の代表空間となる意識原子の集合を、意識空間と呼ぶ。意識空間に属す意識原子は、重複する事が出来ない。意識連鎖の履歴の占有空間は、特異数である。図4参照。

履歴（この場合は意識 $\lambda$ 集合）の代表となる論理原子（この場合は意識原子）が、其の履歴に属しているならば、即ち、空の条件を満たすならば、其の履歴は全体の論理原子を定義しているものではないが、あたかも其れと同じ役割を果たす。空は、其の様な役割を規定するものである。

其の結果、意識連鎖は全体の性質を表す為の存在を定義している事になる。其の様な意識連鎖が幾つも存在すると言う事は、全体を要素とする集合が成立すると言う事である。意識空間は、其の様な性質を持つ空間である。

## 16：論理原子の複写回数の上限数

IDSに属す論理原子が代表空間となる場合に限り、其の論理原子は複写される。論理原子が複写される上限は、其の論理原子が不可知空間からIDSに位相される順位である。複写が上限に達した場合、其の論理原子は代表空間となる事が出来ない。其の場合には、次善の物が選ばれる。

## 17：単元

$\lambda$ 集合の占有空間に可能的に外側で近似する空間観念を有する他の $\lambda$ 集合に属している論理原子を選ぶ作用を単元と呼ぶ。図5参照。

## 18: 確立連鎖

完成した意識  $\lambda$  集合の順列に関して、成立する限りの意識連鎖が定義された後、同じ時点で定義されている認識  $\lambda$  集合の順列の先頭の認識  $\lambda$  集合が選ばれ、且つ、其れに単元を成立させる認識原子が選ばれる。この認識  $\lambda$  集合と、選ばれた単元を成立させる認識原子の関係を確立連鎖と呼ぶ。確立連鎖となる認識  $\lambda$  集合を確立連鎖の履歴、単元を成立させる為に選ばれた認識原子の空間観念を其の確立連鎖の代表空間と呼ぶ。

代表空間となる認識原子は I D S の作用で複写され、I D S の空間観念の外に位相される。確立連鎖の代表空間となる認識原子の集合を、確立空間と呼ぶ。確立空間に属す認識原子は、重複する事が出来ない。

意識連鎖は、成立する限り一度に定義されるのに対し、確立連鎖の場合は、認識  $\lambda$  集合の順列の先頭に並ぶ一個の認識  $\lambda$  集合だけが確立連鎖になる。図 5 参照。

## 19: 開示

確立連鎖が定義されると其の代表空間よりも可能的に外側に近似する代表空間を有する意識連鎖が選ばれる。この作用を開示と呼ぶ。図 6 参照。

## 20: 連想

開示により選ばれる意識連鎖は、其の確立連鎖を成立させたのと同じ認識  $\lambda$  集合の順列の中から自分の履歴に属す意識原子の数と同数の認識原子を有し、且つ、其の占有空間が開示された確立連鎖の占有空間、並びに自分の占有空間よりも可能的に外側に近似する占有空間を有する認識  $\lambda$  集合を選ぶ。この作用を連想と呼ぶ。

## 21: 事象連鎖

連想で選ばれる認識  $\lambda$  集合の代表空間となる認識原子が単元で選ばれる。この場合、選ばれる認識原子は既に確立空間に属すものであったり、或いは、既に事象空間に属すものがあれば、其れ以外のものが選ばれる。選ぶ事が出来なければ、次善の認識原子が選ばれる。この関係で成立する認識  $\lambda$  集合と、選ばれた認識原子の関係を事象連鎖と呼ぶ。事象連鎖の認識  $\lambda$  集合を事象連鎖の履歴、単元を成立させる認識原子の空間観念を其の事象連鎖の代表空間と呼ぶ。

代表空間となる認識原子は I D S の作用で複写され、I D S の空間観念の外に位相される。代表空間となる認識原子の集合を事象空間と呼ぶ。

事象連鎖は、一個の確立連鎖に対応して一個が定義される。事象空間に属す認識原子は、確立空間に属す認識原子、既に事象空間に属す認識原子と重複する事が出来ない。図 6 参照。

## 22: 多重連鎖

事象連鎖が定義されれば、其の事象連鎖の履歴に属す認識原子の数を、例えば  $\alpha$  個とする時、其処から其の事象連鎖の履歴と同じ認識  $\lambda$  集合を除く  $(\alpha^{\alpha} - 1)$  個の集合（以下多重集合と呼ぶ）が、あたかも  $\lambda$  集合と同じ様に存在線上で作られる。同様に、其の事象連鎖の代表空間である認識原子も、多重集合の代表空間を定義するものとして

$(\alpha^{\alpha} - 1)$  個複写される。多重集合と定義された代表空間となる認識原子の関係を多重連鎖と呼ぶ。

この場合の代表空間となる認識原子は I D S の作用で複写され、I D S の空間観念の外に位相される。この代表空間となる認識原子の集合を多重空間と呼ぶ。代表空間となる認識原子の複写が成立しなければ、次善のものが選ばれる。

確立連鎖, 事象連鎖は単元で定義される連鎖となるのに対し, 多重連鎖は其の保証がない連鎖となる。一個の確立連鎖から一個の事象連鎖が作られる関係に対し, 多重連鎖の場合是一个の多重連鎖から  $(\alpha^a - 1)$  個の連鎖が作られる。多重空間に属す認識原子は, 重複している。また, 事象空間に属す認識原子とも重複する事になる。事象連鎖から多重連鎖が定義される作用を多重化と呼ぶ。図 6 参照

### 23: 多重群化と自然群化

本定義の詳細は, 参考文献 A [2] で述べられている。多重連鎖の履歴は確立連鎖並びに事象連鎖の履歴と異なり, 其処に属す認識原子が重複し, 且つ代表空間も重複している。多重空間のこの事態が, 多重空間にこの重複を解消させる為の作用を引き起こす。この作用を多重群化と呼ぶ。多重群化は, 2 個の多重連鎖を対として新たに 1 個の多重連鎖を作り出す作用の事である。新たに 1 個の多重連鎖が作り出されても, 原因となる 2 個の多重連鎖の対は其のまま残る。原因となる 2 個の多重連鎖の対を主従対 (参考文献 A [2]) と呼ぶ。主従対は, 既に定義されている事象連鎖から定義される多重連鎖, 並びに多重群化によって定義された多重連鎖, 特に区別する場合は群化多重連鎖から規則的 (参考文献 A [2]) に決められる。

この作用で, この作用の上述した意図は結果的に解消することが出来ない。確立空間, 事象空間, 意識空間では, 代表空間として存在する論理原子が重複する事がない。其れを摂理とすれば, 多重空間は摂理に反する空間であり, この作用によって更に摂理に反する空間が拡大される。

しかし, この作用により多重連鎖と異なり, 事象連鎖に属す認識原子の数は奇数個となり, 重複せず, 且つ, 其の代表空間が単元を満たす新たな連鎖が誕生する事がある。この連鎖を自然連鎖と呼ぶ。自然連鎖が定義されると, 多重群化の作用は停止する。

自然連鎖の代表空間の認識原子は, 確立空間, 事象空間, 多重空間が定義されるのと同様, 集合を定義する。本研究では, 其れを自然空間と呼ぶ。

自然空間に属す認識原子は, 確立空間, 事象空間と異なり, 多重空間と同様, 重複して定義されてしまう。この事が, 多重空間が摂理に反すると定義されるのと同様, 自然空間も摂理に反すると定義される事になる。

自然空間のこの事態が, 多重空間と同様, この重複を解消させる為の作用を引き起こす。この作用を自然群化 (参考文献 A [2]) と呼ぶ。自然群化の作用は, 概略多重群化の作用と同じである。多重空間は摂理に反する空間であり, 多重群化によって更に摂理に反する空間が拡大されるのと同様, 自然空間も摂理に反する空間は拡大される。

多重群化によって自然連鎖が定義されると, 多重群化の作用は停止し, 自然群化が開始される。

自然群化によって自然連鎖の定義条件外の連鎖, この場合は多重連鎖が定義される事がある。この場合には, 自然群化は停止し, 多重群化が再開される。自然群化と多重群化は, この関係に於いて其の作用を継続する。群化自然連鎖の極限に於いて後述する特異連鎖 (図 8 参照) が誕生する。図 6 参照。

### 24: 自然連鎖

自然連鎖の定義は, 上記 25 参照。本研究で論じている存在とは, 形而上学的な存在の事である。其して, この存在は意識連鎖, 確立連鎖, 事象連鎖, 多重連鎖, 其して自然連鎖として定義される。しかし, 本研究では, われわれの世界は自然連鎖で構成されていると考

える。例えば、われわれの世界の一切の存在は、自然連鎖で構成されている。即ち、われわれが非物質と考える存在も自然連鎖である。例えば、精神、時間、空間などである。われわれが物質と考える存在も、自然連鎖である。例えば、林檎、肉体、路傍の石などである。其して、其の自然連鎖が成立する背景には、意識連鎖、確立連鎖、事象連鎖、其して多重連鎖があると仮説している。われわれの世界の全ての存在と其の振舞いは、本研究では自然群化で規約されていると考える。換言すれば、今日、食事を取るのも自然群化の作用である。

#### 25: 認識連鎖

確立連鎖、事象連鎖、多重連鎖（含む、群化多重連鎖）、自然連鎖（含む、群化自然連鎖）の総称である。認識連鎖の履歴の代表となる認識原子は、其の履歴外の認識原子である。即ち、単元の条件を満たすならば、其の履歴は全体の認識原子の部分を定義している事になる。単元は、其の様な役割を規定するものである。

#### 26: 時間速度の割り込み

新たな時間速度が不可知空間から位相されれば、其の時点で存在線（上記06参照）が消滅し、IDSにおける其の時点までの作用は中断され、IDSの定義は最初からやり直される。これを時間速度の割り込みと呼ぶ。

但し、この場合、IDSの空間観念は定数（上記05参照）故に消滅する事はない。この事により、IDSの空間観念の外にある空間（意識空間、確立空間、事象空間、多重空間、自然空間）も消滅しない。即ち、上記5種の空間はIDSの定義のやり直しとは別に、次第に膨張する事になる。

#### 27: 臨界連鎖

境界原子が代表空間となる意識連鎖を臨界連鎖と呼ぶ。意識連鎖の代表空間は、空を満たす意識原子が選ばれる。選ばれる論理原子がなくなり、次善の論理原子が選ばれる過程で、遂には次善の論理原子もなくなる状況が出現する。図7参照。

#### 28: 特異連鎖

自然群化が継続される事により逐には代表空間の条件を満たす認識原子が見当たらなくなれば、境界原子の空間観念が代表空間となる。この自然連鎖を特異自然連鎖と呼ぶ。図8参照。

#### 29: 意図

特異連鎖の代表空間と臨界連鎖の代表空間が、同じ境界原子で定義される。其の事により、特異連鎖と臨界連鎖は接続される。其れは、特異連鎖は認識連鎖であるので、其の存在の部分の性質を表すものであり、臨界連鎖は意識連鎖であるので其の存在の全体の性質を表す。其の両者が接続するというのは、部分の性質と全体の性質が同等に対応する事を意味する。本研究では、この対応関係を意図が成立すると呼ぶ。この作用は、確立連鎖が定義される事により意識連鎖が開示され、両者の間に対応関係が成立するのと似ている。図9参照。

#### 30: 意識

意図が成立すれば、其の臨界連鎖に属す意識原子の個数と同じ個数の認識原子を要素とする履歴の自然連鎖が、其の特異連鎖を成立させるに至る群化履歴（参考文献A〔2〕参照）の中から選ばれる。この選ばれる自然連鎖が意識を成立させる存在である。選ばれる自然連鎖が其の群化履歴の中に複数個存在すれば、代表空間の小さな方が優先して選ばれる。

。図10参照。

## 31：客体化

意識を成立させる自然連鎖の履歴に属す認識原子のうちの最小の空間観念を代表空間とする自然連鎖が、この意識を成立させる自然連鎖を成立させる群化履歴（参考文献（A〔2〕）を参照）に属していれば、其の自然連鎖はE N Wを成立させる最初の存在として選ばれる。選ばれる自然連鎖を客体化された存在と呼ぶ。

## 32：陳述

客体化された自然連鎖の履歴に属す認識原子のうちの客体化された自然連鎖の代表空間よりも大きな空間観念を有する認識原子の空間観念を代表空間とする自然連鎖が、この意識を成立させる自然連鎖を成立させる群化履歴（参考文献（A〔2〕）を参照）に属していれば、其の自然連鎖が選ばれる。この選ばれる自然連鎖を陳述された存在と呼ぶ。陳述された存在は、上文の条件を満たせば繰り返し選ばれる。

## 33：客体化，陳述の停止

上記33，34で選ばれた自然連鎖の履歴が多重連鎖の履歴と一致すれば、其の客体化，または陳述は其処で停止する。選ばれる自然連鎖がこの状況に至れば、この意識を成立させる自然連鎖に端を発したE N Wの構築作用は終了する。

## 34：記憶と同化

客体化，あるいは陳述で選ばれる自然連鎖の履歴に属す認識原子の全てについて陳述が成立すれば、この場合、其の陳述を行う自然連鎖には同化が成立する。其の自然連鎖の履歴に属す認識原子の部分が陳述されれば、この場合、其の陳述を行う自然連鎖には記憶が成立する。記憶とは、われわれが感じられる存在の事である。同化とは、われわれが感じられない存在の事である。

## 35：転位と回帰

存在線に於ける境界原子は、不可知空間（U）から新たな時間速度が移送される事により、認識原子に遷移する事がある。この関係を転位と呼ぶ。逆に、認識原子が境界原子に遷移する事を回帰と呼ぶ。本研究では時系列的に時間速度の大なる方が過去（祖先）、小なる方が未来と位置づけられるので、前者の転位は外延的作用で時系列的関係を成立させる。其れに対し、後者の回帰は内包的作用で時系列的に可逆となる。

他方、PRDに於いて、其処に配置されるSF間には、先に配置されるSF程、未来の役割を担い、後に続く程、過去を担う。TDMが風船に譬えた様に時系列的関係を逆にして成立する構造である事を想起すれば、PRDも其れに準拠する事は必然である。其の事からR3D、R3Mの経路作用は過去から未来への遷移を意味し、R2C、R3Cの経路作用は未来から過去への遷移を意味する。特に、同期状態を保証し得ない直列に並ぶ3個のSFの両端のSFに於いて、同期を求められる始点となる主語に対して、本研究では上述の関係から其の主語を転位（境界）、回帰（逆転位）と呼ぶ。

## 第六章 纏め

本研究では、ソフトは、其れを写すプログラムを構成する全ステートメントを決定する為の公理的規則を求める理論として論じられるべきであると主張する。

この問題に挑む為にはソフトを工学的思考法から切り離し、ソフトを形而上学的に定義する事から始めなければならないと主張する。

其して、ソフトウェアエンジニアリングが遭遇しているジレンマを克服する為には、これ



ら主張が研究者、技術者には欠く事が出来ない心構えであると主張する。

本研究の着想は、データ構造中心設計の先駆けであるワーニエによる方法 [参考文献 A [11] 参照] が発表される以前に遡る。

其の意味で、本研究は長期に渡る研究成果と言う事が出来る。其して、其れは3つの時期に分けて回顧する事が出来る。初期では主に論文の収集調査、中期では主にソフトのプロダクの調査、其して初期並びに中期の間に80余のソフト開発の指導に携り、其れらが本研究の背景を成す基盤となっている。後期では、本研究の狙いであるソフトウェアの定義と其れをプログラム言語で定義する方法（開発方法論）の為の論考が行われた。

この間、外部のソフトウェアエンジニアリングの世界では、様々な開発のアプローチ（機能構造化、データ構造化、オブジェクト化）が提案されていた。其して、学術的にはコンピューター工学、ソフトウェアサイエンス、情報科学、情報工学等の曖昧な差別化が行われていた。其して、様々な団体が生まれ、様々な概念、技術用語の提案が図られていた。其の事は、問題の解決とは別に今も行われている様である。

本研究の成果は、第三章（3.5）で述べている様に、今風に言えば、普遍的な最小のクラスを定義した事である。これは、ソフトウェアを普遍的な最小のクラスの集合で定義出来る事を示す。其の事によって、第一章（1.1）で述べているソフトウェアを開発する場合に遭遇するジレンマの原因である曖昧さ、恣意性を改善的に回避する事が出来る。これは、これまでの様々なアプローチでは改善する事が困難な共通の問題であった。

其のことに於いて、本研究の成果はこれまでの様々なアプローチの共通の問題を改善する統一的な開発方法論に帰結していると言う事が出来る。

即ち、本研究で言うプログラムの公式は、開発案件に属す最小のオブジェクトクラスが、22種の主語の種別（オブジェクト種別）、26種のベクトルの作用の種別（メソッドの種別）、其して、ベクトルの作用を定義する69種の規則で公式的に定義出来る事から、これは、プログラムを決定する標準理論と呼ぶ事が出来るであろう。

この様な成果は、これまでのソフトウェアエンジニアリングの世界と一線を画した事により到達出来たと思われる。其れは、この種の研究はひとりで行わない限り其の目標に到達する事が困難であると思われる点が多々あるからである。本研究に30余年を要しているが、同一人格者が30人いれば1年で到達出来ると考える事も出来る。しかし、同一人格者を30人養成する事は不可能である。他方、本研究を通じて確信される事は、ソフトの研究を惑う事なく進めれば、認識を公共化させると言う工学的世界の本質性の壁に遭遇する。其れは、ソフトの世界では誰もが求める命題ではあるが、現実的に殆ど無し得ない問題なのである。其の事において、この課題に挑もうとするならば、観察点を変える発想の変換が求められる。其れを多くの人に求めるのは、世代の風潮からすれば殆ど不可能な事である。

この様な状況の中で、本研究は其の結果だけを多くの人々に提供すると言う事を目的とした。其れが、本研究が長期に及んだ原因である。既述の事から、ソフトは哲学的世界観を必要とする世界の様に思われる。其して、ソフト世界の傾向は、本研究で言う様なプログラムの構造論を研究する視点を見失っている様に思われる。もし、工学が定義から始まるのに対し、哲学は定義に始まり定義に終わるという事を想起すれば、論理哲学のWittgenstein, モナロロジのLeibniz, エチカのSpinoza, 公理不定性のGodel, 其して集合論のCantorの様な人々が今おられれば、ソフトは今とは違う世界観の基で実現されているのではないかと思われる。

現状の様々なアプローチは、図 268 の様に開発プロセスを 3 段階に区別している。其して、開発案件と設計作業との関係は、開発案件を満たす本質的なアルゴリズムの内容よりも、開発案件と設計との妥当性を立証する方法を確定する事が出来ない為に、両者の間のプロセス管理に力点が置かれてしまっている。また、設計とプログラム構築との関係は、設計とプログラムの構築との検証性を立証する方法が出来ない為に、プログラムが設計事情を満たしているかどうかの検証のプロセス管理に力点が置かれてしまっている。即ち、作業のプロセス管理が重要視され、あたかもソフトウェアの品質が其れによって決まるかの風潮が高まっている。この風潮は、ソフトウェアの本質性とは区別されるべきもので、ソフトウェアの進化とも区別されるべきものである。

其の為に、開発案件、設計、プログラムの構築のプロセスが一元的に捉えられない状況になっている。この問題に端を発して、ソフトウェアエンジニアリングを構成する概念、方法論、用語等の定義が其の最終ゴールであるプログラムとリンク出来ない事が、ソフトウェアエンジニアリングの世界が混迷を続けている原因であり、且つ、学問的にも成熟し得ないのは、この混迷が背景にある様に思われる。

本研究は、第一章（1. 4）の論考の道筋で述べている様に、プログラムを定義するアルゴリズムを開発案件の論理から可能な限り切り分けて定義可能とする事に挑戦している。其して、プログラムの 1 ステートメントを開発要件から求める為の思考法が、ベクトルの定義規則として確立されている。其して、其れが最小のクラスを普遍的に定義する事を可能にし、ベクトルの第 1 規約が其のベクトルの妥当性を述定し、第 3 規約が検証性を述定している事から、これまでのソフトウェアエンジニアリングのアプローチに対し、本研究の成果は図 269 の様に纏められる。

本研究のプログラムの公式は、上図の通り従来のアプローチが規定する工程（開発案件、設計、プログラムの構築）を一元化させる役割を果たしている。この事が、各工程を其の最終ゴールの視点から整理可能とする効果を作り出している。

#### 6. 1 本研究の成果が齎すソフト開発作業上の効果

開発案件は機能的に示されても、其の言表手段が言葉である限り、序論でも触れている様に、曖昧さが生じるのは当然である。其れは、ソフトウェアの世界においては原理的に除去出来るものではない。其の結果、開発案件として言表したい事とプログラムとの間には Gap が生じる。これも必然とせねばならない。このような問題をこれまでのアプローチで克服する事は、困難な作業である。其して、個人の天分に依存するしかないと言うのが実情である。

本プログラムの公式のアルゴリズムは、第三章（3. 6）で述べている様に開発案件を可能な限り機械的に規約する。其の事が上記のやむを得ないと考えられる実情を改善させる事が出来る。

#### 6. 2 本研究の成果が齎すソフト開発思想のパラダイム変革の可能性

本プログラムの公式の効果として、次の様なソフト開発思想のパラダイム変革を齎す可能性が期待出来る。

- (01) 自動プログラミングを可能にする。
- (02) プログラムの検証方法の考え方を変える。
- (03) 開発案件の捉え方の考え方を変える。
- (04) ソフト開発費用の市場評価を確立させる事が出来る。

- (05) ソフト開発期間の短縮化（リアルタイム開発）を実現させる。
- (06) ソフトを其のシステム環境（ハード、OS、ミドルソフト）から独立させる。
- (07) EUC（エンドユーザーコンピューティング）を実現させる。

## 6. 3 結論

この公式で開発されたシステムの実績（表9）は、この3年間で約31件である。其の内訳は、ビジネスソフトで小さなものもあれば大きなものもある。実績がビジネスソフトに片寄っているのは他の分野、例えば制御系、基本ソフトの分野で開発の機会が得られない事に拠る。

本研究を通じて感じられる事は、ソフトにはひとびとの思考的自由を奪う力を持つ側面があり、其れ故、真に普遍的な事柄以外は、単に技巧的な取り決めを世界基準とするべきではないと考える。其の様な事態から得られる効果は、将来大きな禍根の原因となると考えるからである。本研究の成果が其の様な便宜的な風潮を打破し、健全な未来に向かう礎石のひとつになればと願うものである。

### 【産業上の利用可能性】

#### 【0019】

上記で詳細に説明したように、本発明はソフトウェア産業のみならず、ソフトウェアを利用する全産業に極めて有用な効果をもたらす。

### 【図面の簡単な説明】

#### 【0020】

【図1】IDSの模式図である。

【図2】論理原始（Vi）の空間観念を表す概念図である。

【図3】占有空間の概念を説明するための概念図である。

【図4】空の連鎖の概念を説明するための概念図である。

【図5】単元の連鎖の概念を説明するための概念図である。

【図6】連鎖の成立過程の概念を説明するための概念図である。

【図7】臨界連鎖の概念を説明するための概念図である。

【図8】特異連鎖の概念を説明するための概念図である。

【図9】意図の概念を説明するための概念図である。

【図10】意識の概念を説明するための概念図である。

【図11】ENWの模式図である。

【図12】ENWを構成する存在の関係を概念的に説明するための概念図である。

【図13】ENWの再定義を概念的に説明するための概念図である。

【図14】L4のためのPSの構造とその規則を概念的に説明するための概念図である。

【図15】L2のためのPSの構造とその規則を概念的に説明するための概念図である。

【図16】L3のためのPSの構造とその規則を概念的に説明するための概念図である。

【図17】TDMの模式図である。

【図18】SFを概念的に説明するための概念図である。

【図19】PRDを概念的に説明するための概念図である。

【図20】パレット連鎖関数の定義規則を概念的に説明するための概念図である。

【図21】パレット関数の定義規則を概念的に説明するための概念図である。

【図22】I2の作用と領域との関係を概念的に説明するための概念図である。

【図23】I2の領域の関係を概念的に説明するための概念図である。

【図24】I2の作成単位を概念的に説明するための概念図である。

- 【図 2 5】 O 4 の領域の関係を概念的に説明するための概念図である。。
- 【図 2 6】 O 4 の作成単位を概念的に説明するための概念図である。
- 【図 2 7】 本理論の標準の骨格を概念的に説明するための概念図である。
- 【図 2 8】 主語の種別定義を概念的に説明するための概念図である。
- 【図 2 9】 主語の属性を概念的に説明するための概念図である。
- 【図 3 0】 パレットに属すベクトルの種別を概念的に説明するための概念図である。
- 【図 3 1】 S 4 の主語となる領域の種別を概念的に説明するための概念図である。
- 【図 3 2】 ベクトル種別と領域数を概念的に説明するための概念図である。
- 【図 3 3】 L y e e B E L T を概念的に説明するための概念図である。
- 【図 3 4】 経路制御テーブルの例である。
- 【図 3 5】 ベクトルの型一覧表である。
- 【図 3 6】 ベクトルの型一覧表である。
- 【図 3 7】 ベクトルの型一覧表である。
- 【図 3 8】 L y e e B E L T の例である。
- 【図 3 9】 ベクトルの型の例である。
- 【図 4 0】 ベクトルの型の例である。
- 【図 4 1】 ベクトルの例である。
- 【図 4 2】 ベクトルの例である。
- 【図 4 3】 パレット関数の型の例である。
- 【図 4 4】 パレット関数の型の例である。
- 【図 4 5】 パレット関数の型の例である。
- 【図 4 6】 パレット関数の型の例である。
- 【図 4 7】 パレット関数の型の例である。
- 【図 4 8】 パレット関数の型の例である。
- 【図 4 9】 パレット関数の型の例である。
- 【図 5 0】 パレット関数の型の例である。
- 【図 5 1】 パレット連鎖関数の型の例である。
- 【図 5 2】 パレット連鎖関数の型の例である。
- 【図 5 3】 パレット連鎖関数の型の例である。
- 【図 5 4】 パレット連鎖関数の型の例である。
- 【図 5 5】 パレット連鎖関数の型の例である。
- 【図 5 6】 パレット連鎖関数の型の例である。
- 【図 5 7】 パレット連鎖関数の型の例である。
- 【図 5 8】 パレット連鎖関数の型の例である。
- 【図 5 9】 パレット連鎖関数の型の例である。
- 【図 6 0】 パレット連鎖関数の型の例である。
- 【図 6 1】 パレット連鎖関数の型の例である。
- 【図 6 2】 パレット連鎖関数の型の例である。
- 【図 6 3】 パレット連鎖関数の型の例である。
- 【図 6 4】 パレット連鎖関数の型の例である。
- 【図 6 5】 パレット連鎖関数の型の例である。
- 【図 6 6】 定義規則図 1 である。
- 【図 6 7】 定義規則図 2 である。
- 【図 6 8】 定義規則図 3 である。
- 【図 6 9】 定義規則図 4 である。
- 【図 7 0】 定義規則図 5 である。
- 【図 7 1】 定義規則図 6 である。
- 【図 7 2】 定義規則図 7 である。
- 【図 7 3】 定義規則図 8 である。
- 【図 7 4】 定義規則図 9 である。

- 【図 75】 定義規則図 10 である。
- 【図 76】 定義規則図 11 である。
- 【図 77】 定義規則図 12 である。
- 【図 78】 定義規則図 13 である。
- 【図 79】 定義規則図 14 である。
- 【図 80】 定義規則図 15 である。
- 【図 81】 定義規則図 16 である。
- 【図 82】 定義規則図 17 である。
- 【図 83】 定義規則図 18 である。
- 【図 84】 定義規則図 19、20 及び 21 である。
- 【図 85】 定義規則図 22、23 及び 24 である。
- 【図 86】 定義規則図 25 である。
- 【図 87】 定義規則図 26 である。
- 【図 88】 定義規則図 27 である。
- 【図 89】 定義規則図 28 である。
- 【図 90】 定義規則図 29 である。
- 【図 91】 定義規則図 30 である。
- 【図 92】 定義規則図 31 である。
- 【図 93】 定義規則図 32 である。
- 【図 94】 定義規則図 33 である。
- 【図 95】 定義規則図 34 である。
- 【図 96】 定義規則図 35 である。
- 【図 97】 定義規則図 36 である。
- 【図 98】 全ベクトルの型: 001 のコードである。
- 【図 99】 全ベクトルの型: 001 のコードである。
- 【図 100】 全ベクトルの型: 002 のコードである。
- 【図 101】 全ベクトルの型: 002 のコードである。
- 【図 102】 全ベクトルの型: 003 のコードである。
- 【図 103】 全ベクトルの型: 003 のコードである。
- 【図 104】 全ベクトルの型: 004 のコードである。
- 【図 105】 全ベクトルの型: 004 のコードである。
- 【図 106】 全ベクトルの型: 004 のコードである。
- 【図 107】 全ベクトルの型: 005 のコードである。
- 【図 108】 全ベクトルの型: 005 のコードである。
- 【図 109】 全ベクトルの型: 005 のコードである。
- 【図 110】 全ベクトルの型: 006 のコードである。
- 【図 111】 全ベクトルの型: 006 のコードである。
- 【図 112】 全ベクトルの型: 006 のコードである。
- 【図 113】 全ベクトルの型: 007 のコードである。
- 【図 114】 全ベクトルの型: 007 のコードである。
- 【図 115】 全ベクトルの型: 008 のコードである。
- 【図 116】 全ベクトルの型: 008 のコードである。
- 【図 117】 全ベクトルの型: 009 のコードである。
- 【図 118】 全ベクトルの型: 009 のコードである。
- 【図 119】 全ベクトルの型: 009 のコードである。
- 【図 120】 全ベクトルの型: 010 のコードである。
- 【図 121】 全ベクトルの型: 010 のコードである。
- 【図 122】 全ベクトルの型: 010 のコードである。
- 【図 123】 全ベクトルの型: 011 のコードである。
- 【図 124】 全ベクトルの型: 011 のコードである。

- 出証特 2 0 0 4 - 3 0 9 9 1 9 2

- 出証特 2 0 0 4 - 3 0 9 9 1 9 2

- 【図 2 2 5】全ベクトルの型: 0 5 4 のコードである。
- 【図 2 2 6】全ベクトルの型: 0 5 4 のコードである。
- 【図 2 2 7】全ベクトルの型: 0 5 5 のコードである。
- 【図 2 2 8】全ベクトルの型: 0 5 5 のコードである。
- 【図 2 2 9】全ベクトルの型: 0 5 6 のコードである。
- 【図 2 3 0】全ベクトルの型: 0 5 6 のコードである。
- 【図 2 3 1】全ベクトルの型: 0 5 7 のコードである。
- 【図 2 3 2】全ベクトルの型: 0 5 7 のコードである。
- 【図 2 3 3】全ベクトルの型: 0 5 7 のコードである。
- 【図 2 3 4】全ベクトルの型: 0 5 8 のコードである。
- 【図 2 3 5】全ベクトルの型: 0 5 8 のコードである。
- 【図 2 3 6】全ベクトルの型: 0 5 8 のコードである。
- 【図 2 3 7】全ベクトルの型: 0 5 9 のコードである。
- 【図 2 3 8】全ベクトルの型: 0 5 9 のコードである。
- 【図 2 3 9】全ベクトルの型: 0 5 9 のコードである。
- 【図 2 4 0】全ベクトルの型: 0 6 0 のコードである。
- 【図 2 4 1】全ベクトルの型: 0 6 0 のコードである。
- 【図 2 4 2】全ベクトルの型: 0 6 0 のコードである。
- 【図 2 4 3】全ベクトルの型: 0 6 1 のコードである。
- 【図 2 4 4】全ベクトルの型: 0 6 1 のコードである。
- 【図 2 4 5】全ベクトルの型: 0 6 2 のコードである。
- 【図 2 4 6】全ベクトルの型: 0 6 2 のコードである。
- 【図 2 4 7】全ベクトルの型: 0 6 2 のコードである。
- 【図 2 4 8】全ベクトルの型: 0 6 3 のコードである。
- 【図 2 4 9】全ベクトルの型: 0 6 4 のコードである。
- 【図 2 5 0】全ベクトルの型: 0 6 5 のコードである。
- 【図 2 5 1】全ベクトルの型: 0 6 6 のコードである。
- 【図 2 5 2】全ベクトルの型: 0 6 7 のコードである。
- 【図 2 5 3】全ベクトルの型: 0 6 8 のコードである。
- 【図 2 5 4】全ベクトルの型: 0 6 9 のコードである。
- 【図 2 5 5】全ベクトルの型: 0 7 0 のコードである。
- 【図 2 5 6】全ベクトルの型: 0 7 1 のコードである。
- 【図 2 5 7】全ベクトルの型: 0 7 2 のコードである。
- 【図 2 5 8】全ベクトルの型: 0 7 3 のコードである。
- 【図 2 5 9】全ベクトルの型: 0 7 4 のコードである。
- 【図 2 6 0】全ベクトルの型: 0 7 5 のコードである。
- 【図 2 6 1】全ベクトルの型: 0 7 6 のコードである。
- 【図 2 6 2】全ベクトルの型: 0 7 7 のコードである。
- 【図 2 6 3】全ベクトルの型: 0 7 8 のコードである。
- 【図 2 6 4】全ベクトルの型: 0 7 9 のコードである。
- 【図 2 6 5】全ベクトルの型: 0 8 0 のコードである。
- 【図 2 6 6】本研究の論考の道筋を示す概念図である。
- 【図 2 6 7】本研究のアルゴリズムを模式的に表す概念図である。
- 【図 2 6 8】現状のソフトウェアエンジニアリングのアプローチを模式的に示す概念図である。
- 【図 2 6 9】本研究によるソフトウェアエンジニアリングを模式的に示す概念図である。

【符号の説明】

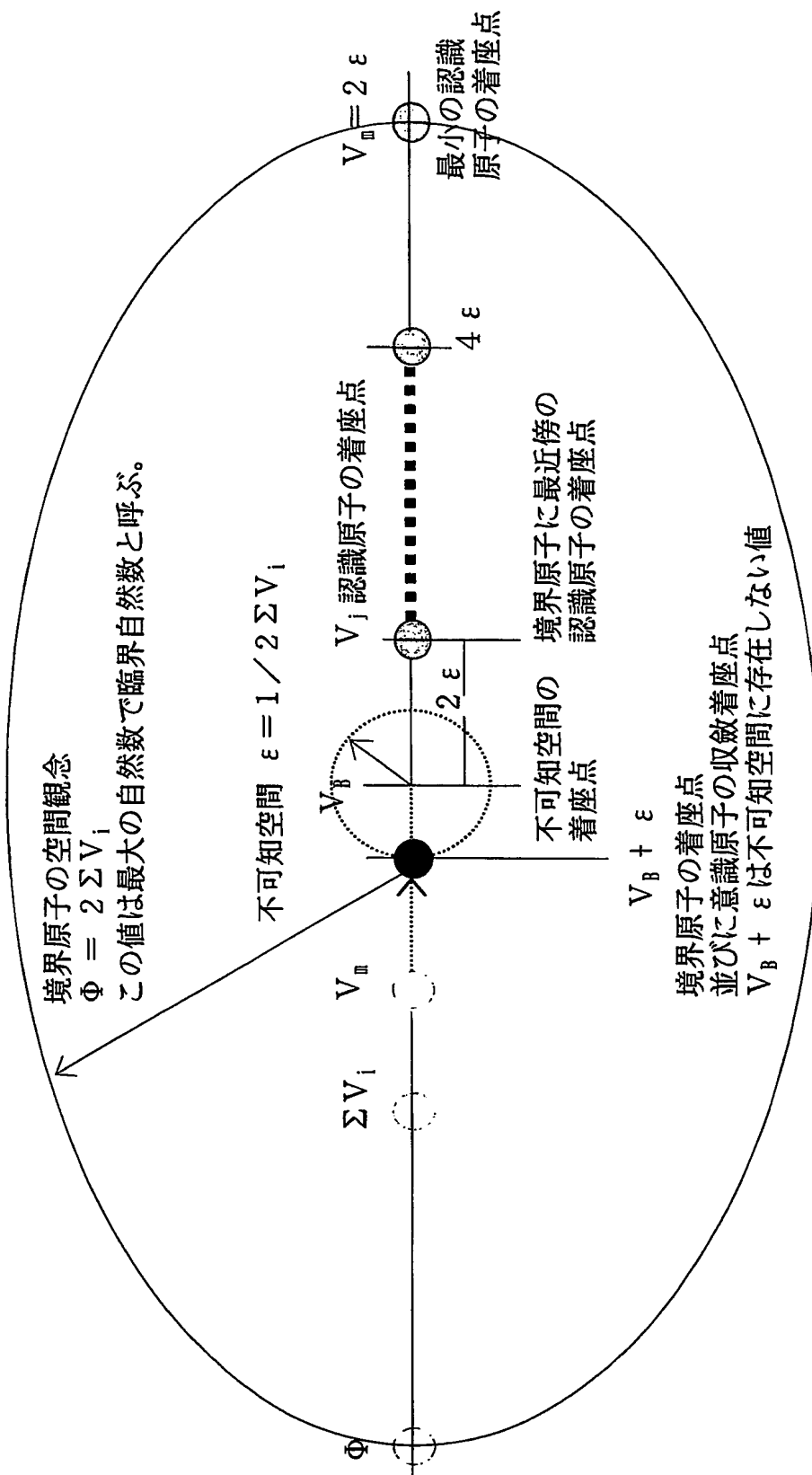
【0 0 2 1】

PRD 処理経路図

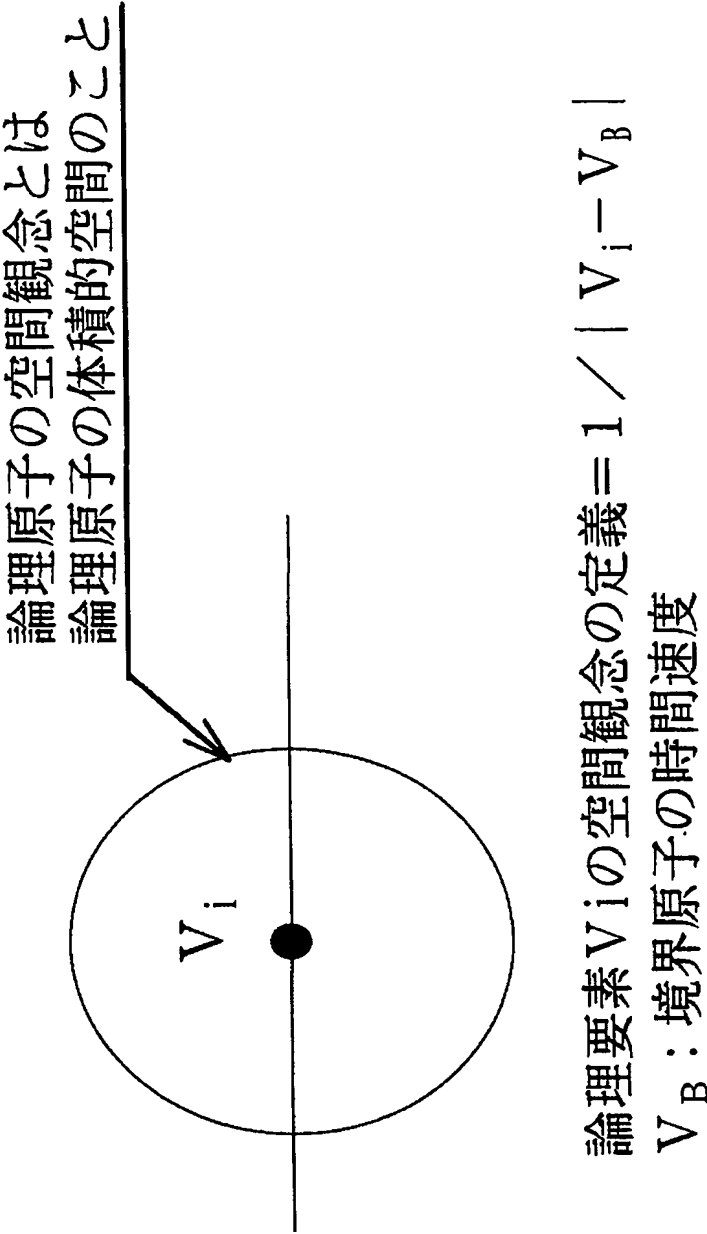


SF シナリオ関数

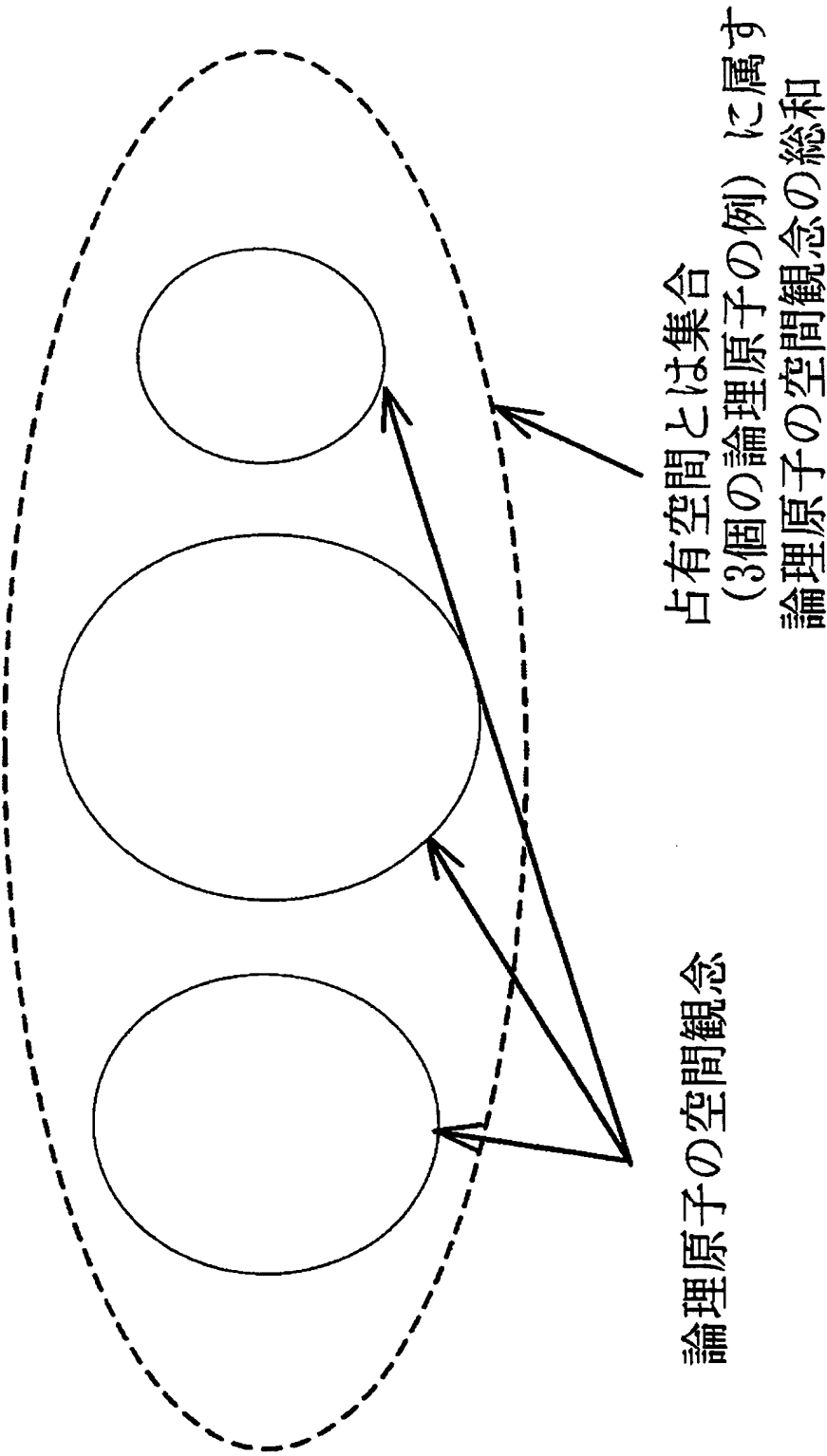
【書類名】 図面  
【図 1】



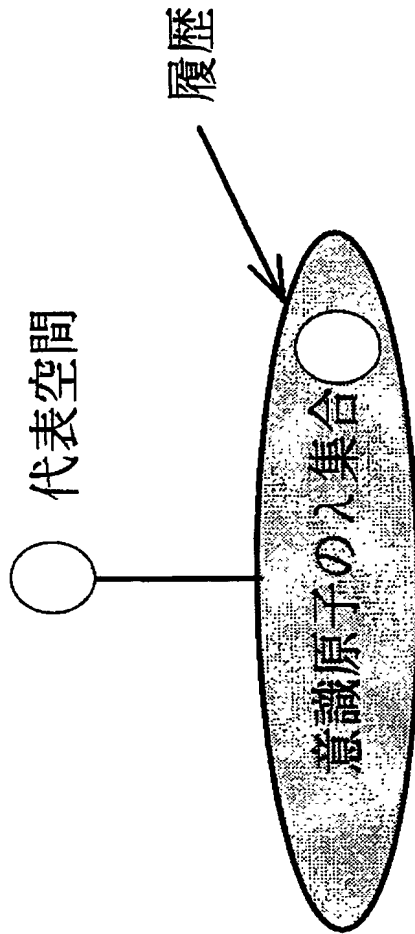
【図 2】



【図 3】

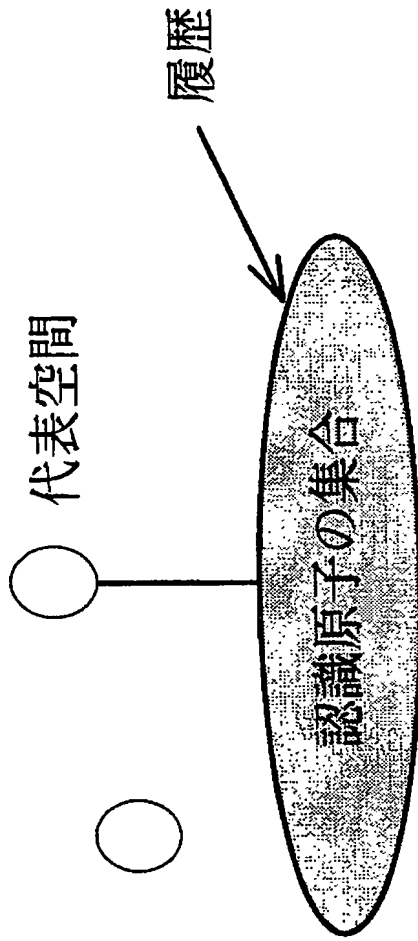


【図 4】



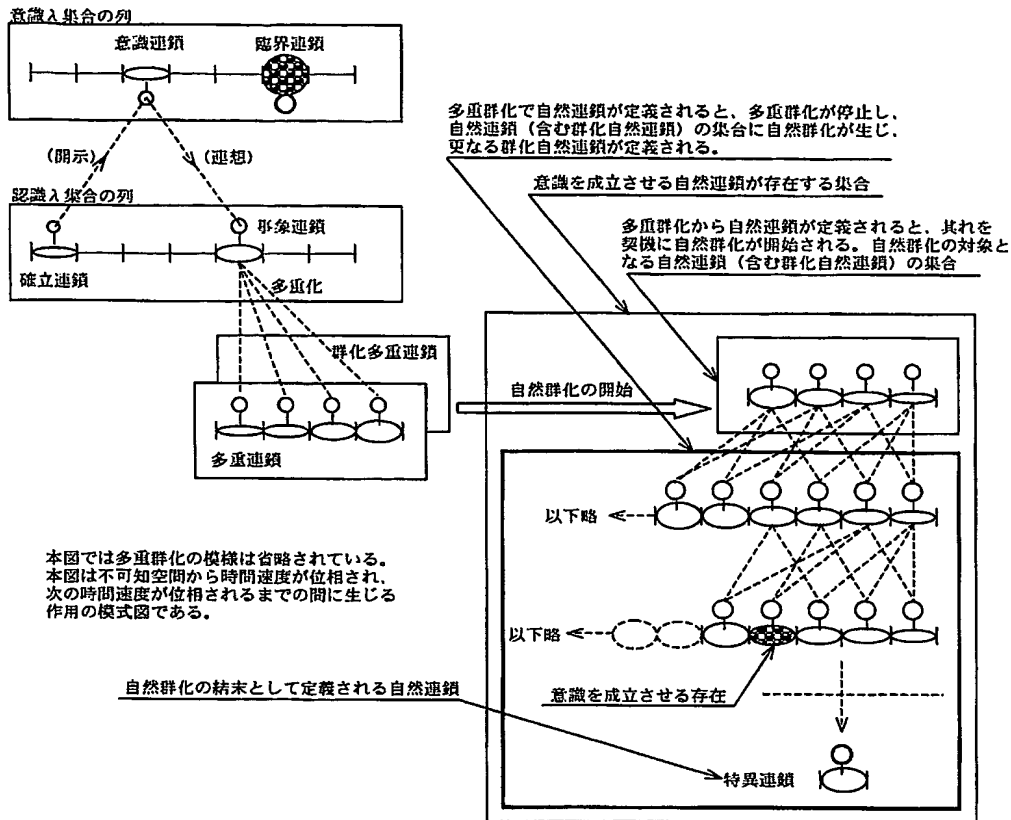
- ①空とは代表空間となる論理原子がその履歴に属す事である。  
 ②空で成立する連鎖には意識連鎖がある。  
 ③空で成立する連鎖の履歴は内包的である。

【図 5】



- ①単元とは代表空間となる論理原子がその履歴に属さない事である。  
 ②単元で成立する連鎖には確立連鎖、事象連鎖、自然連鎖がある。  
 ③単元で成立する連鎖の履歴は外延的である。

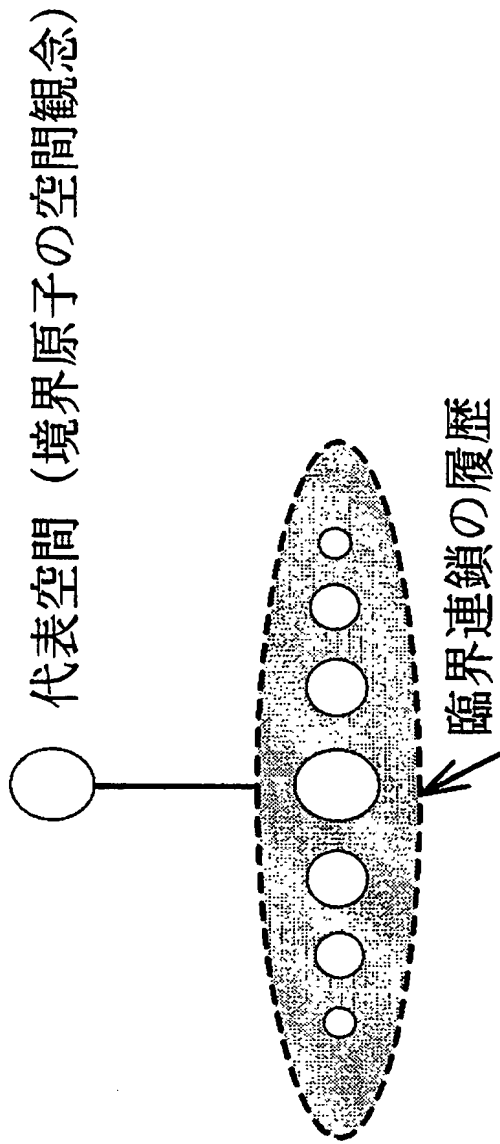
【図 6】



本図では多重群化の模様は省略されている。  
本図は不可知空間から時間速度が位相され、  
次の時間速度が位相されるまでの間に生じる  
作用の模式図である。

群化自然連鎖の集合の矢印は、自然群化が行われる順序を示す。

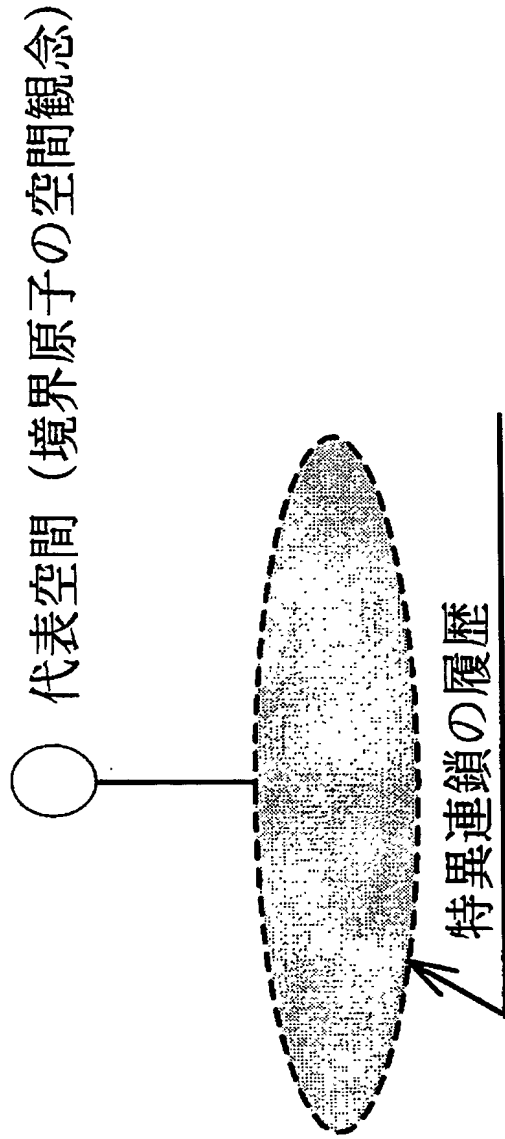
【図 7】



本図は臨界連鎖の例示的模式図である。  
代表空間 (境界原子) は、臨界連鎖の履歴に属している (空)。



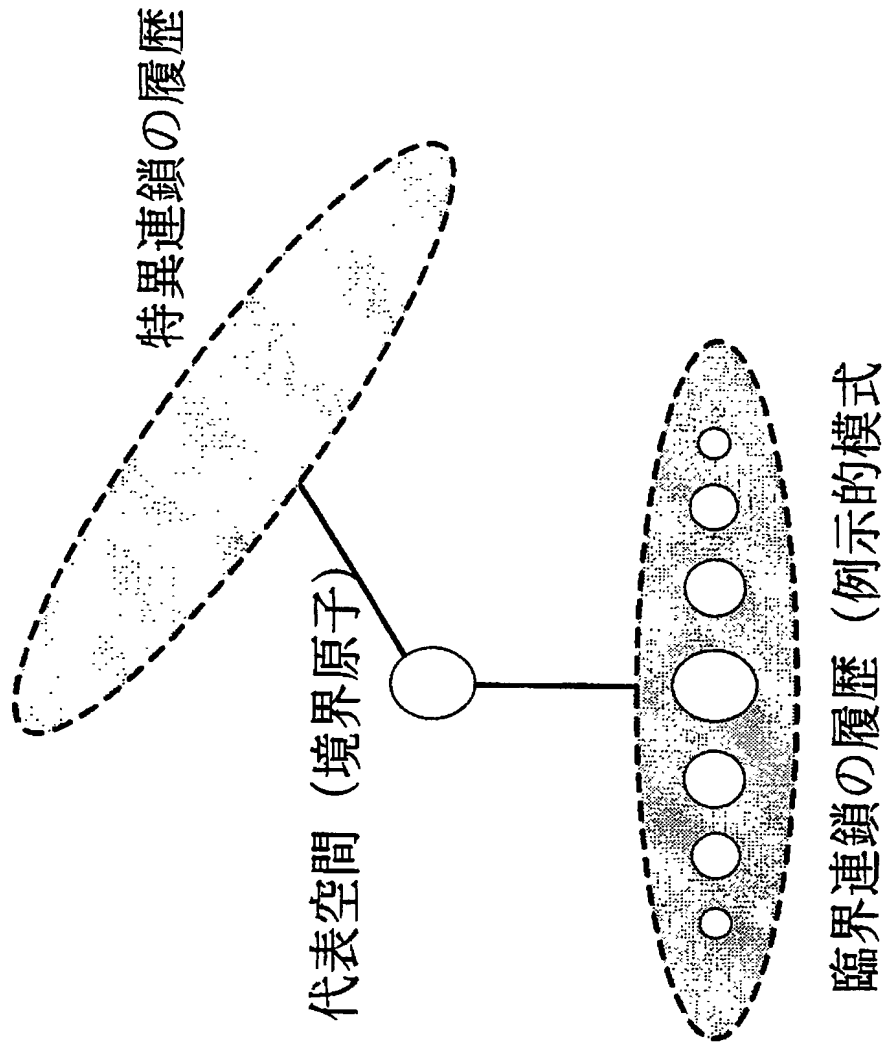
【図 8】



この集合には認識原子の総数が属すと予想される。

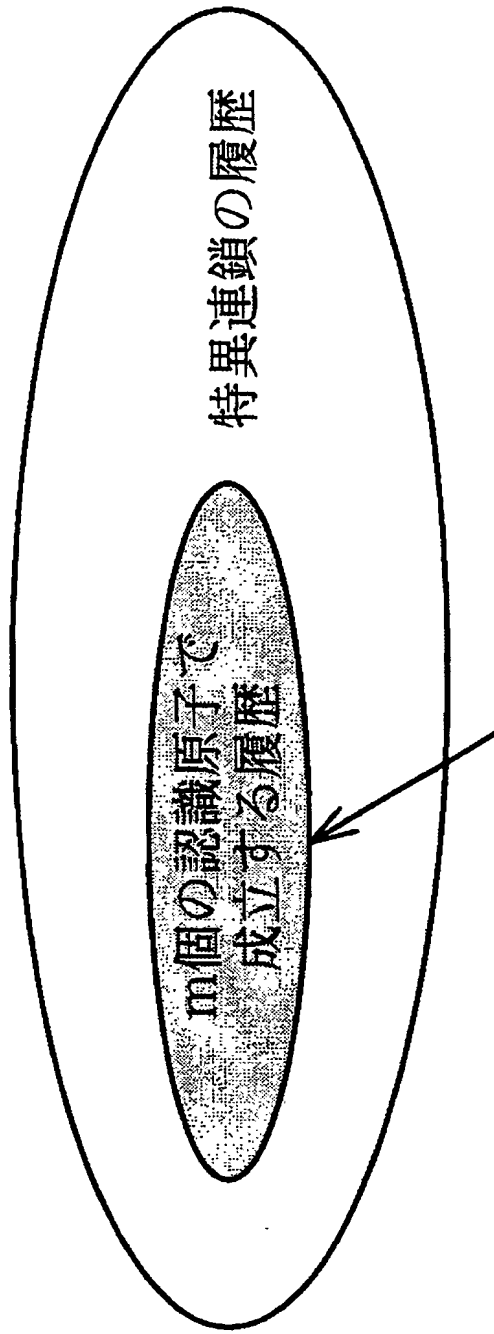
本図は特異連鎖の例示的模式図である。  
代表空間 (境界原子) は、特異連鎖の履歴に属していない (単元)。

【図9】



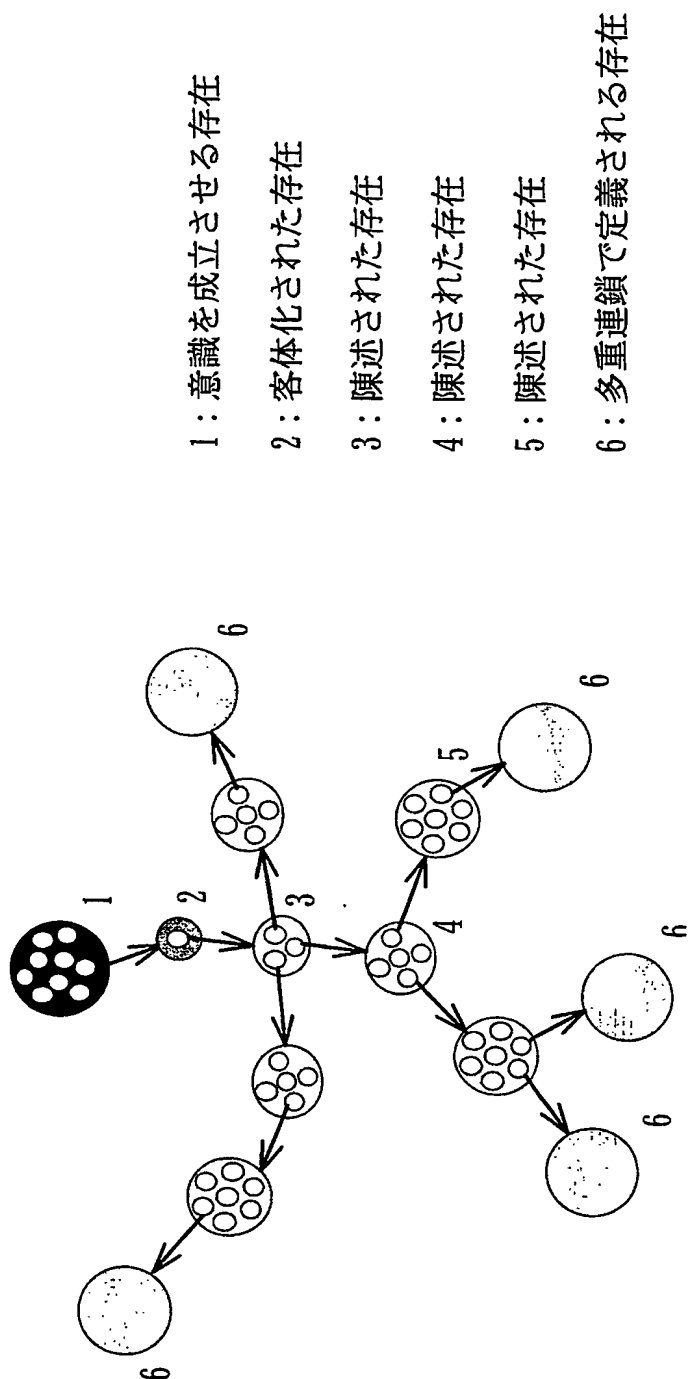
これは、部分と全体がブリッジされた状態である。

【図 10】



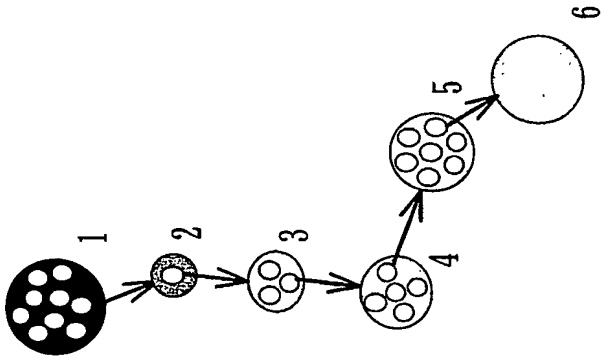
意識を成立させる自然連鎖の履歴 (m個の認識原子で成立する)  
意識は、意図が成立する元で定義される概念である。  
其の場合の臨界連鎖の履歴に属す意識原子の数がm個である。

【図 11】



【図 1 2】

- (1) 2を自分とした場合の祖先・子孫の関係  
1：2の祖先（記憶不能）  
2：自分  
3：2の子孫
- (2) 3を自分とした場合の祖先・子孫の関係  
2：3の祖先  
3：自分  
4：3の子孫
- (3) 5を自分とした場合の祖先・子孫の関係  
4：5の祖先  
5：自分  
6：自分の子孫（記憶不能）



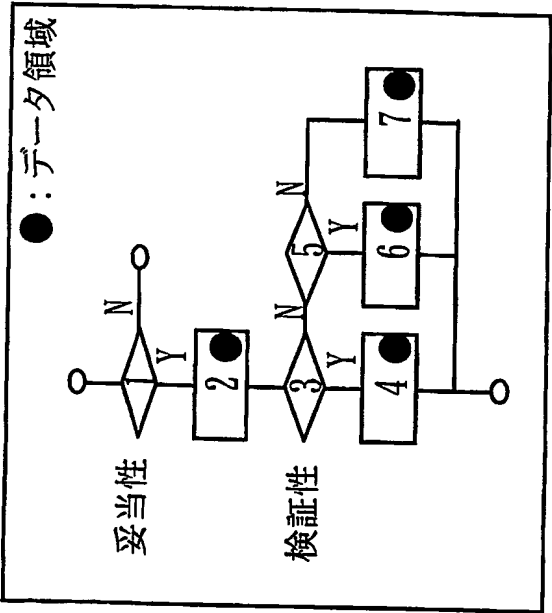
(図 1 1 の部分)





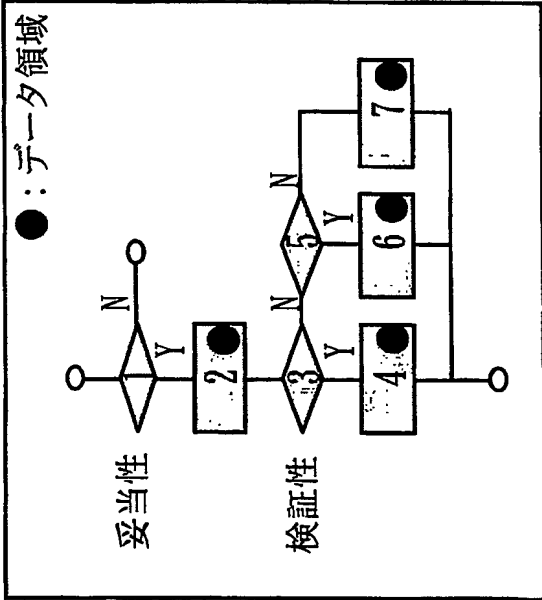
【図 1 5】

N0	規約名	P S の定義規則（確立連鎖の性質）
1	第1規約	自分の子孫の妥当性判定を述語化する
2	第2規約	自分の存在を子孫を用いて述語化する
3	第3規約	第2規約の述語化を検証する述語化を行う
4	第4規約	自分の存在を述語化する
5	第5規約	第2規約の述語化の検証性を述語化する
6	第6規約	自分の子孫が存在である事を述語化する
7	第7規約	自分の子孫が不備である事を述語化する



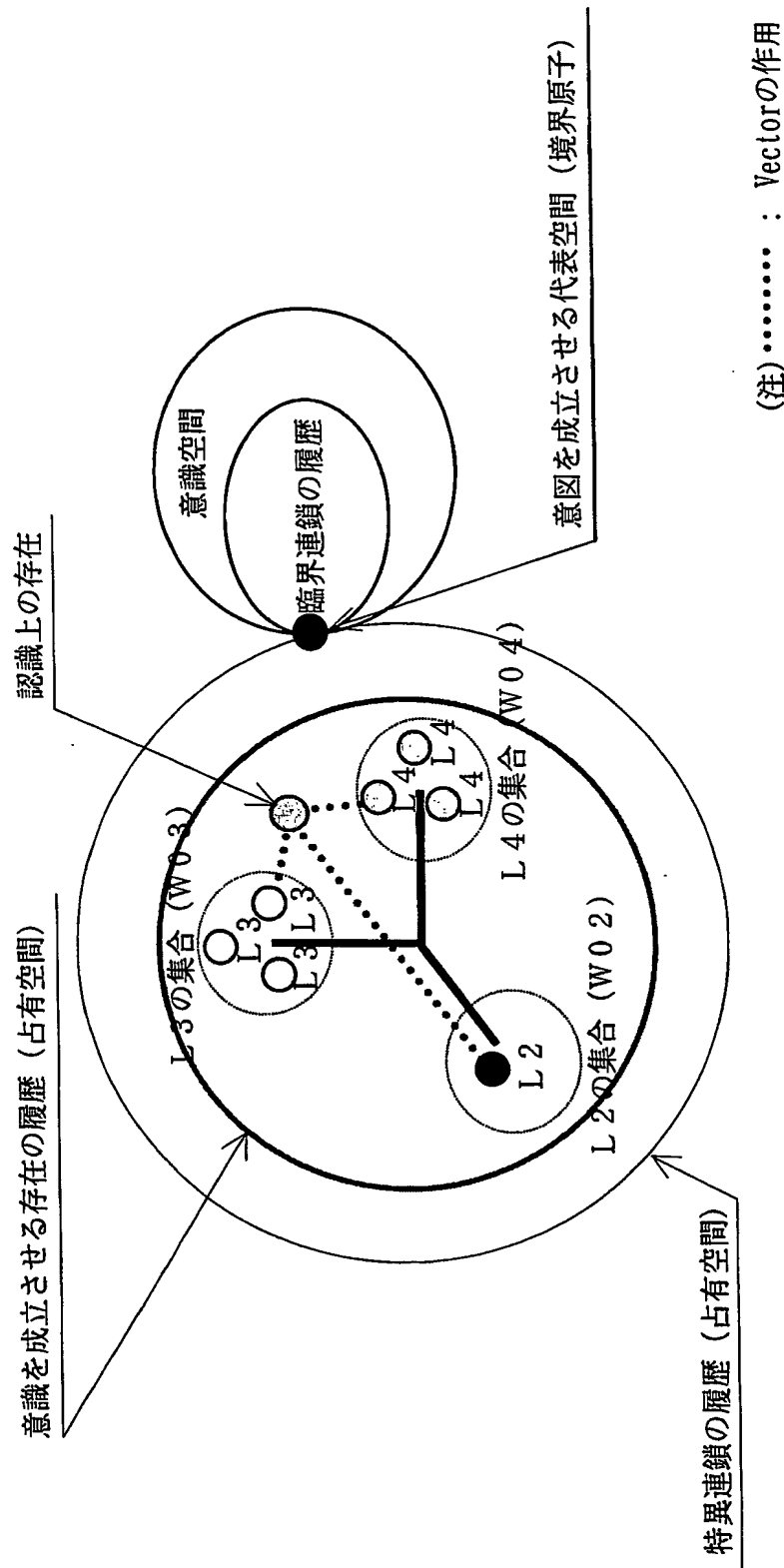


【図 1 6】



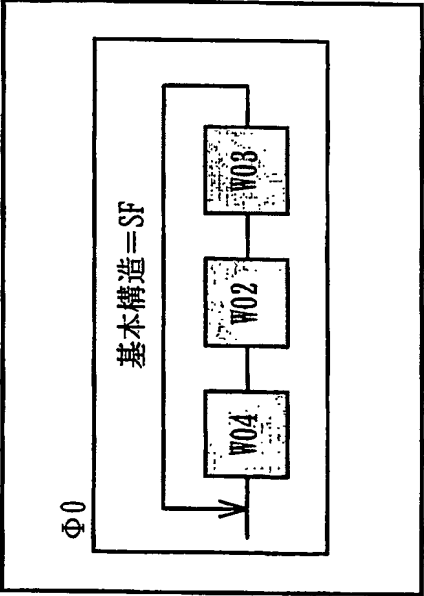
N0	規約名	PSの定義規則（多重連鎖の性質）
1	第1規約	自分の子孫の妥当性判定を述語化する
2	第2規約	述語化不要
3	第3規約	自分の存在を子孫を用いて述語化出来るかどうかを述語化する
4	第4規約	自分の存在を述語化する
5	第5規約	第3規約の述語化の検証性を述語化する
6	第6規約	自分の子孫が存在である事を述語化する
7	第7規約	自分の子孫が不備である事を述語化する

【図 17】



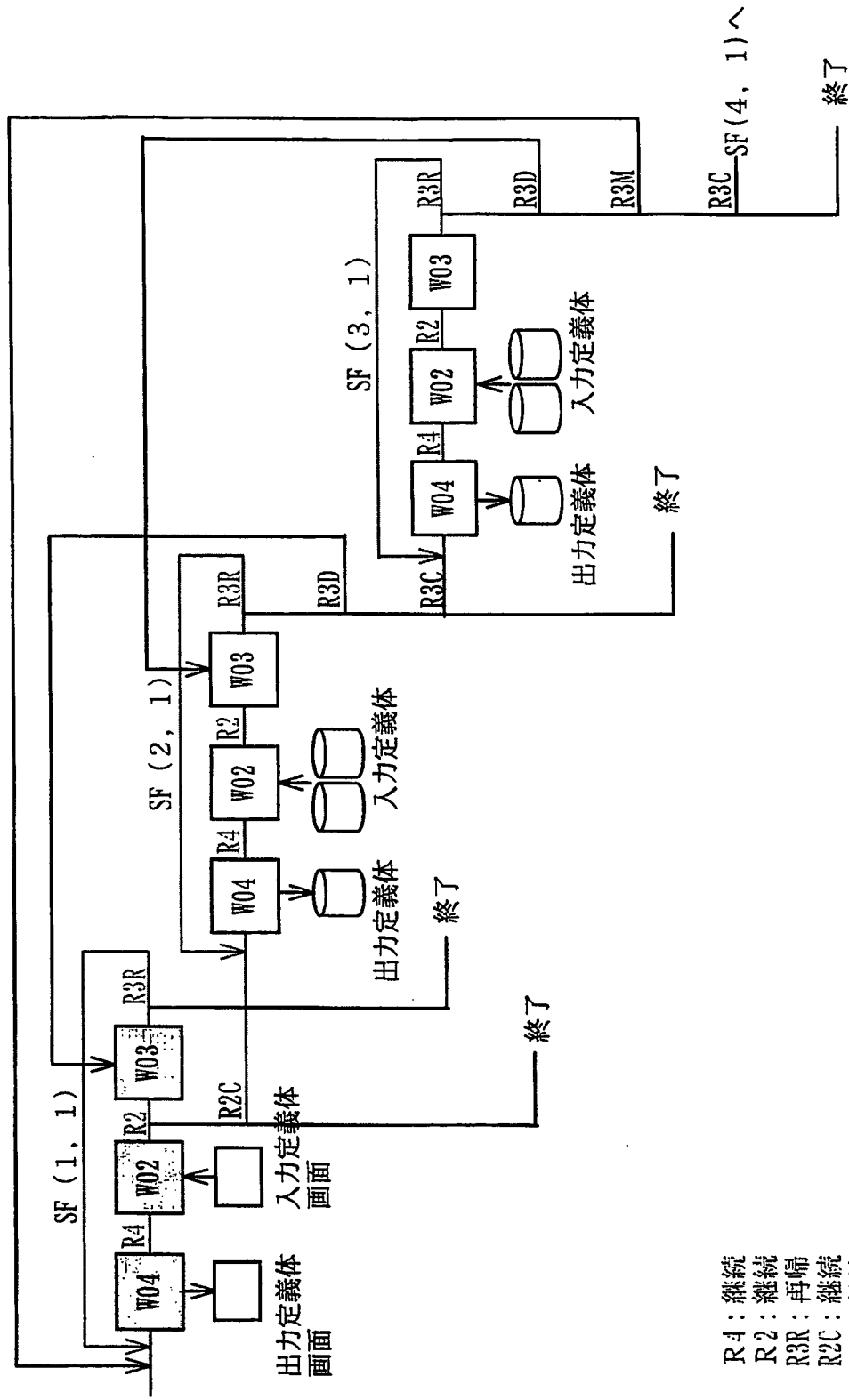
【図 1 8】

パレット関数				ベクトル
$\Phi 0$	(SF1)	$\Phi 0$ (1)	W04パレット	L4、O4、S4、R4
			W02パレット	L2、I2、R2C、R2
			W03パレット	L3、R3R
	(SF2)	$\Phi 0$ (2)	W04パレット	L4、O4、S4、R4
			W02パレット	L2、I2、R2C、R2
			W03パレット	L3、R3R
	(SF3)	$\Phi 0$ (3)	W04パレット	L4、O4、S4、R4
			W02パレット	L2、I2、R2C、R2
			W03パレット	L3、R3R



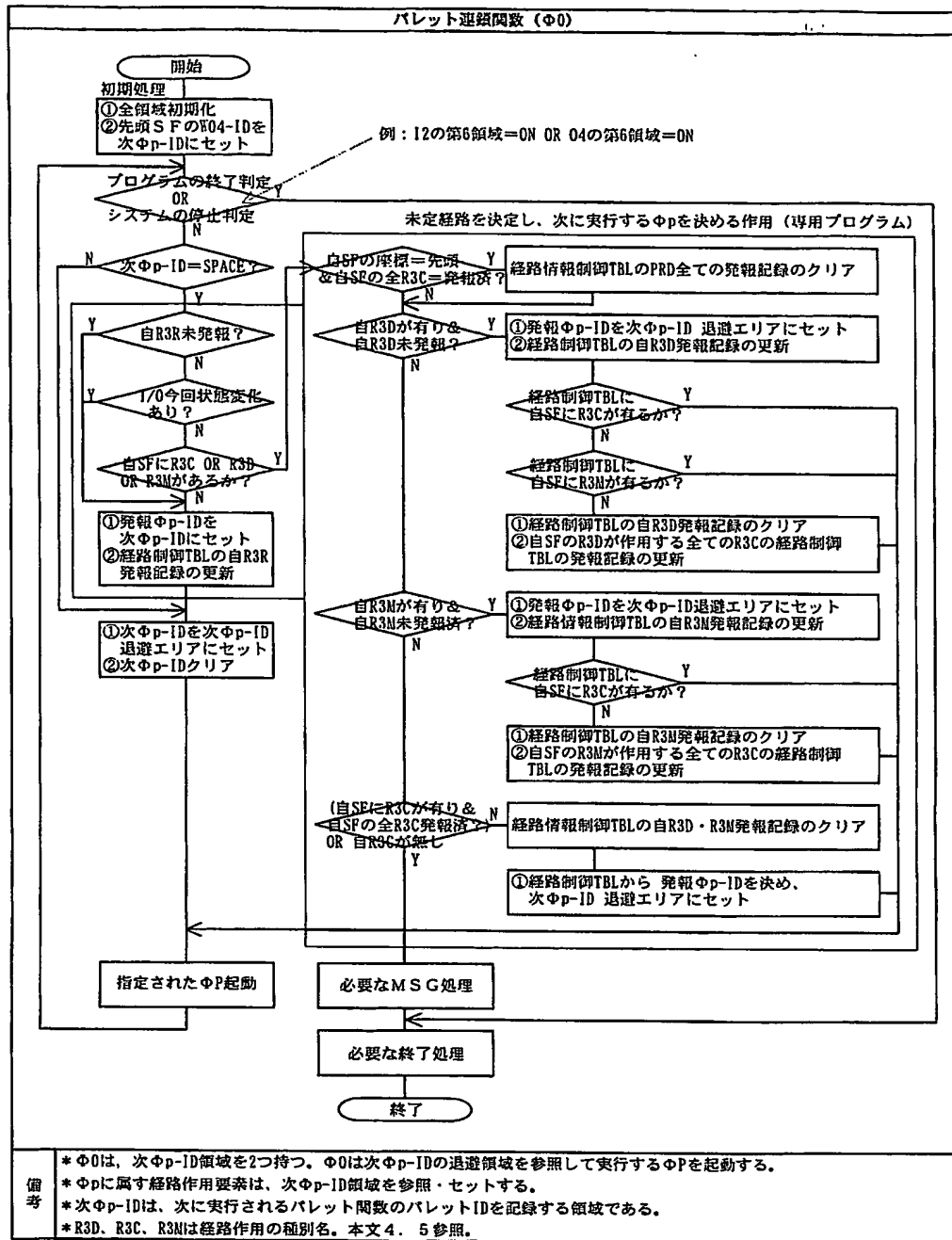
【図 19】

PRD 1

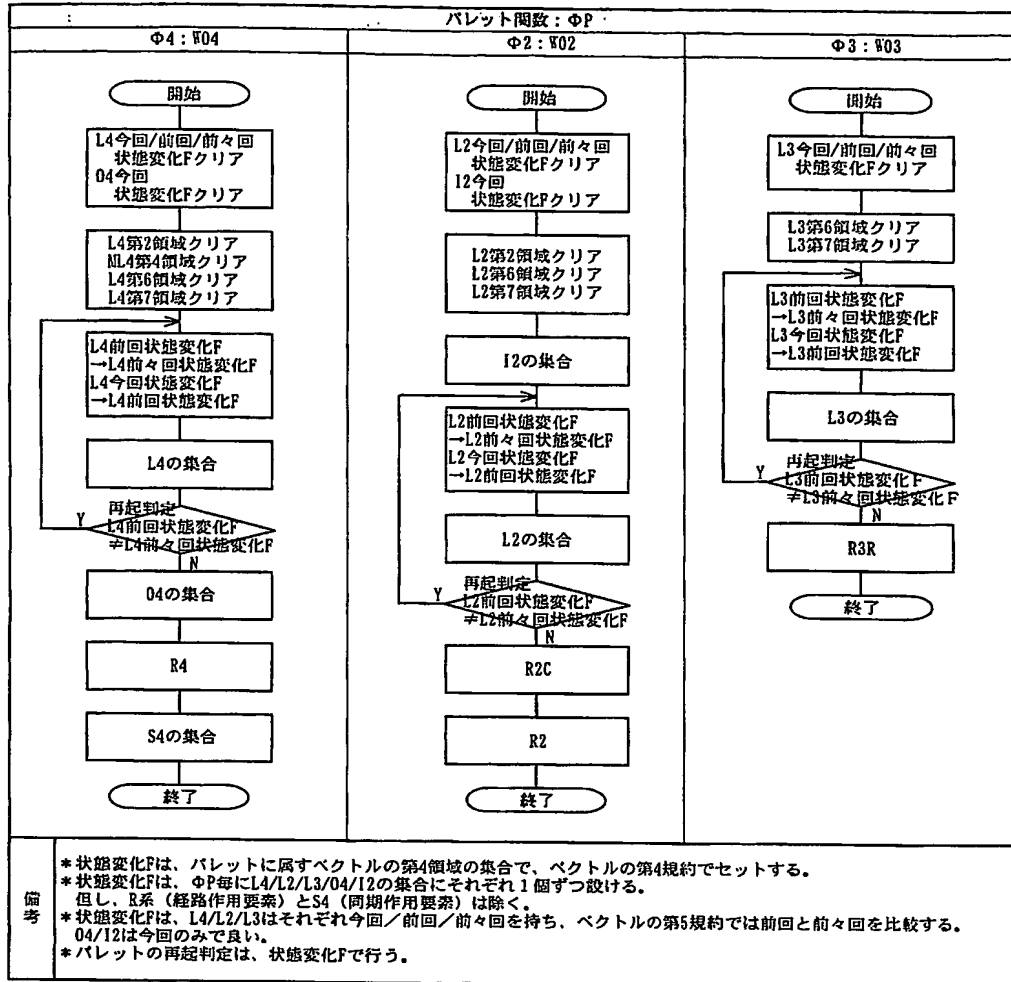


\*SF (4, 1) 以降は、SF (3, 1) と同様

【図 20】

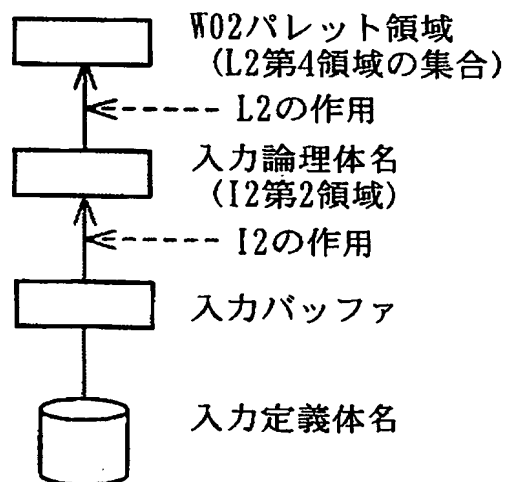


【図 2 1】



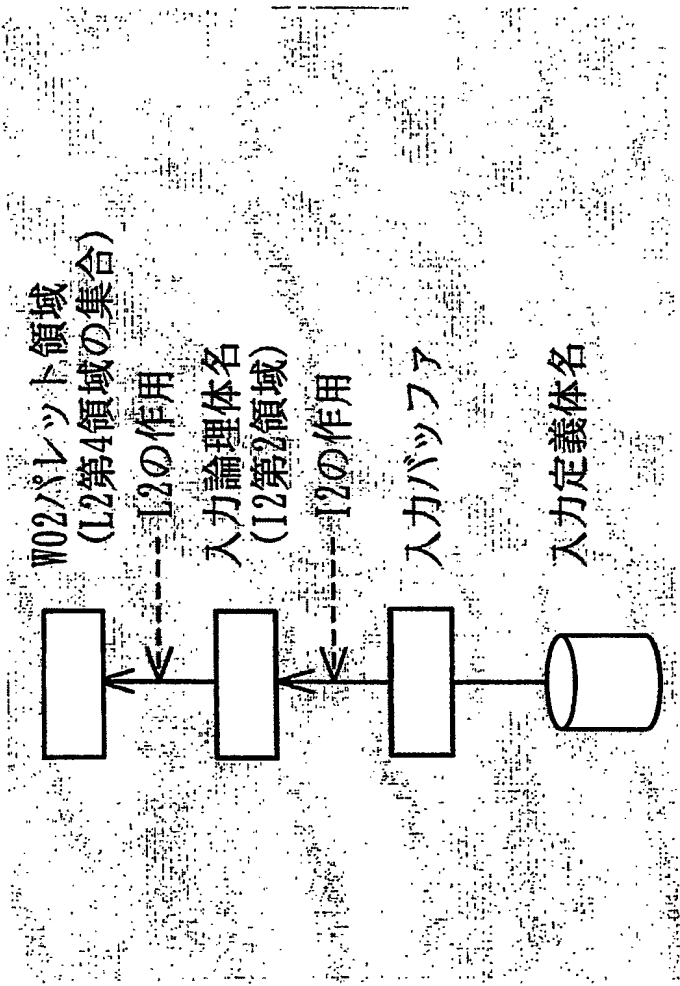
219・パレット関数の命令規則

【図 2 2】



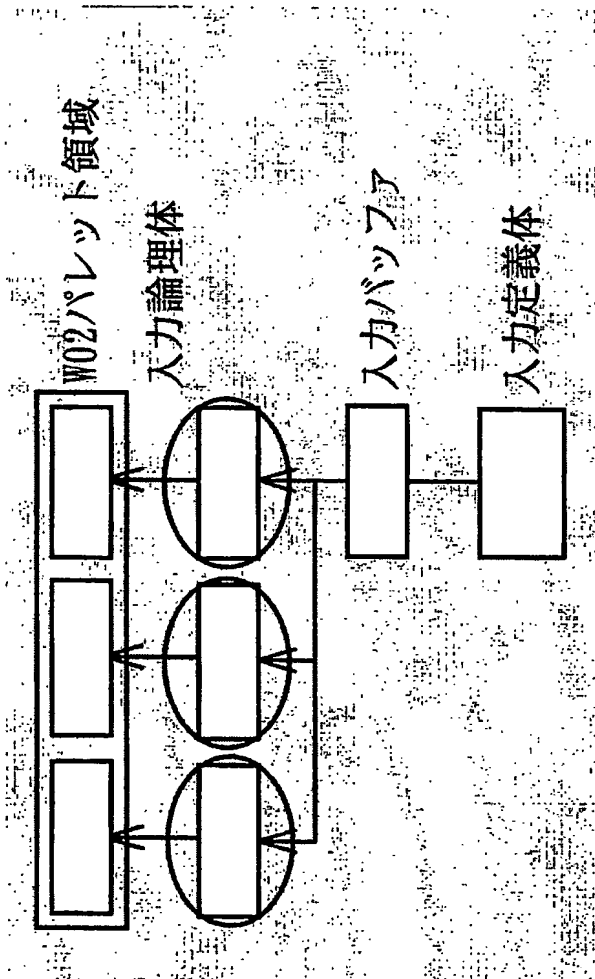
備考

I2の第2領域と、L2第4領域  
の集合はそれぞれ別個に定  
義される。



【図 24】

備考  
左記○で囲まれた入力論理体毎に、I2を作成する。

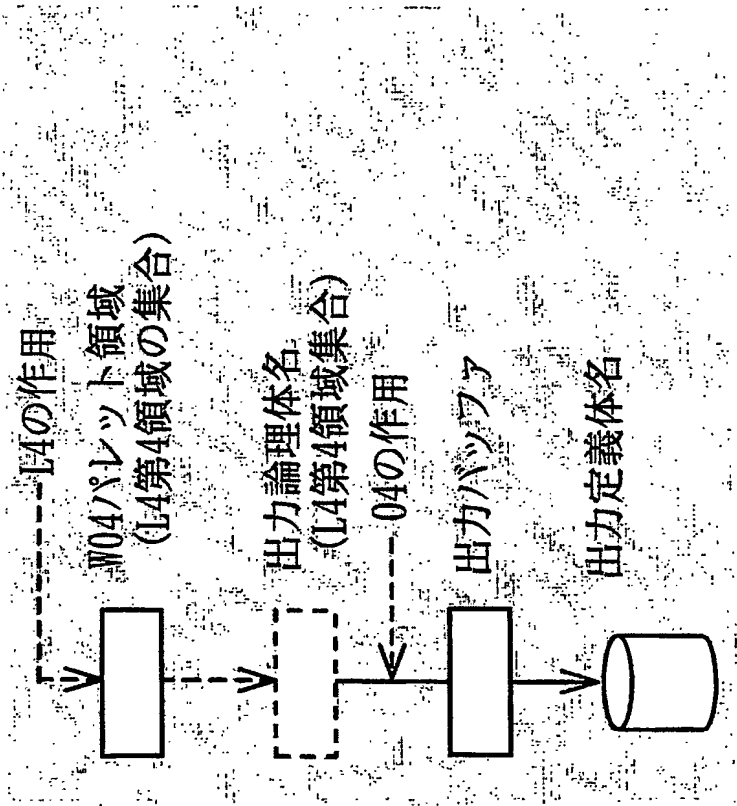




備考

【図 25】

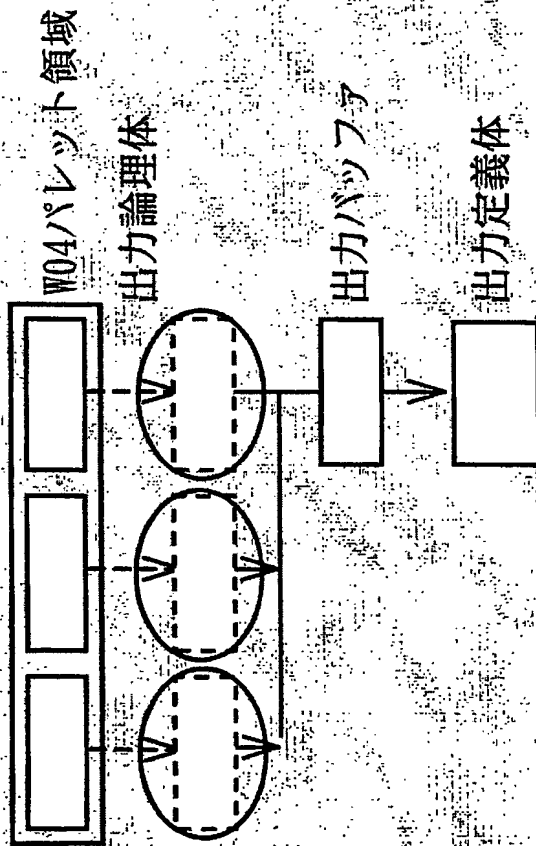
04の第2領域は、L4第4領域  
の集合と同じである



【図 26】

備考

左記○で囲まれた出力論理体毎に、04を作成する。



【図 27】

IAK: 入力アクセスキー、ICK: 入力処理条件キー  
OAK: 出力アクセスキー、OCK: 出力処理条件キー

S4の主部となる領域														
ベクトル 識別	属性				業務				S4の主部となる領域					
	入力	出力	配列	境界	正規 処理	発生単位	総理体		IAK	ICK	OAK	OCK	L3	L3
						発生単位	IAK	ICK	OAK	OCK	L3	OCK	L3	単位
正規	L2	O	O	O	01									48
		O	O	O	02									49
		O	O	O	03									51
		O	O	O	04									55
		O	O	O	05									56
		O	O	O	06									57
		O	O	O	07									58
		O	O	O	08									59
		O	O	O	09									60
		O	O	O	10									61
		O	O	O	11									62
		O	O	O	12									63
		O	O	O	13									64
		O	O	O	14									65
		O	O	O	15									66
		O	O	O	16									67
		O	O	O	17									68
		O	O	O	18									69
入力 総理体	L2	O	-	-				19						
	L3	O	-	-			20	21						
出力 総理体	L4	O	-	-										
	L3	O	-	-										
	L4	O	-	-										
総路	R4	-	-	-										
	R2C	-	-	-										
	R2	-	-	-										
	R3R	-	-	-										

\*表中の番号は、定義範囲を示す。

【図 28】

項番	主語	定義
1	正規	開発案件に明示される名詞。 例えば、画面、ファイル、帳票に属す項目名。
2	K	正規主語でベクトルを定義する事が出来ない場合に派生させる主語。 例えば、集計処理に用いる補助領域。
3	M	開発案件に属すべき名詞（正規）でありながら、それを認めない状況で存在する主語。
4	アクセスキー	入力コマンドの為のアクセスキーを定義するベクトルの主語。
5	入力作用	入力コマンドの為の処理条件キーを定義するベクトルの主語。
6	論理体	入力コマンドを定義するベクトルの主語。
7	アクセスキー	出力コマンドの為のアクセスキーを定義するベクトルの主語。
8	出力作用	出力コマンドの為の処理条件キーを定義するベクトルの主語。
9	論理体	出力コマンドを定義するベクトルの主語。
10	パレット	SFに属すW04、W02、W03の識別子。
11	領域	S4（ベクトル）を定義する為の主語。

【図 29】

項番	属性	定義
1	入力	W02に配置される論理体（入力論理体と呼ぶ）に属す主語の属性を「入力」と呼ぶ。
2	出力	W04に配置される論理体（出力論理体と呼ぶ）に属す主語の属性を「出力」と呼ぶ。
3	配列	1個の主語が識別化された複数の領域を有する場合、その主語の属性を「配列」と呼ぶ。
4	等価	1個の主語が、異なる複数の論理性を有する場合、その主語は等価（性）を有すると呼ぶ。
5	境界	異なるSF（異なる意識）を同期させる為に用いる主語の論理性が成立する場合、その主語は境界（性）を有すると呼ぶ。この属性は、PRD（図17）の構造から認識的に判別する事が出来る。

【図 30】

項番	種別	所属			分類	役割	ベクトル名		工学的概念	備考
		W04	W02	W03						
1	L4	○			1	論理	論理要素		計算式 付与情報	端点の始点の正しさを求める論理
2	L2		○		1				計算条件	無条件で成立してゐる始点
3	L3			○	1					端点の存在を確認する論理
4	I2		○		2	作用	作用要素	入力	読み込み	
5	O4	○			2			出力	書き込み	
6	S4	○			3		同期作用要素		クリア	非空処理
7	R4	○			3	作用	経路作用要素	継続	処理の連鎖	操作情報から設定
8	R2C		○		1			継続		
9	R2		○		3			継続		
10	R3R			○	3			再帰		

【図 3 1】

項番	ΦP	S 4 の主語	
		ペクトル種別	領域種別
1	W04	L4	第4
2		04	第4
3	W02	L2	第4
4		I2	第2
5			第4
6	W03	入力アクセスキー	第4
7		入力処理条件キー	第4
8		出力アクセスキー	第4
9		出力処理条件キー	第4
10		単語	第4

\* 表記されている入力アクセスキーの領域、入出力処理条件キー、I2、04の領域を制御BOXと呼ぶ。

【図 3 2】

項番	主語		Vector							
	種別 (列)	配列 (列)	属性		第2領域 個数	第4領域 個数	Vector 個数	第2領域 個数	第4領域 個数	Vector 個数
			等価	境界						
1	正規				1	1	1	-	1	1
2		○			m	m	1	-	m	1
3			○		-	-	-	-	共通	n ≥ 2
4		○	○		-	-	-	-	m, 共通	n ≥ 2
5				○	-	-	-	-	2	1
6			○	○	-	-	-	-	2, 共通	n ≥ 2
7		○	○	○	-	-	-	-	2, m	1
8		○	○	○	-	-	-	-	2 (m-共通)	n ≥ 2
9	K				-	1	1	-	1	1
10			○		-	1	1	-	共通	n ≥ 2
11		○			-	1	1	-	m	1
12		○	○		-	1	1	-	m, 共通	n ≥ 2
13	N				-	-	-	-	1	1
14			○		-	-	-	-	共通	n ≥ 2
15		○			-	-	-	-	m	1
16		○	○		-	-	-	-	m, 共通	n ≥ 2
17	入力論理体: IAK, ICK				-	-	-	-	-	-
18	出力論理体: OAK, OCK				-	-	-	-	-	-
19	入力論理体				1	1	1	-	-	-
20	出力論理体				-	-	-	(=14第4)	-	-
21	R4				-	-	-	-	共通	1
22	R2C				-	-	-	-	共通	1
23	R2				-	-	-	-	共通	1
24	R3R				-	-	-	-	共通	1

\* IAK: 入力アクセスキー、ICK: 入力処理条件キーを示す。  
\* OAK: 出力アクセスキー、OCK: 出力処理条件キーを示す。  
\* nは等価性を有する単語の論理性の種別数を示す。  
\* mは配列を有する単語の配列定数 (列と行の積数) を示す。  
\* L4第4領域の2は、正規と境界の2つの領域が存在する事を示す。  
\* L3第2領域は、業務上使用しない為、不要。



【図 33】

項番	主語	定義
1	名称	表 2 で定義される主語の種別に従って主語の名称を定義する。
2	識別子	上記 1 で定義される主語の名称のプログラム名を定義する。
3	入出力区分	入力コマンドが対象とする主語を入力、出力コマンドが対象とする主語を出力を定義する。
4	主語種別	ベクトル (ベソッド) を定義する為に、表 2 の主語の種別を表 1 の主語の種別として再定義する。
5	属性の組合わせの型	ベクトルを定義する為に、表 1 の主語の種別を表 3 の主語の属性を基に再定義する。
6	等価種別数	主語に成立する等価性 (表 3) の個数を定義する。
7	等価識別子	等価種別毎に識別子を定義する。
8	座標	主語を修飾する PRD・SF・パレット・論理体の固有情報 (座標) を定義する。
9	論理体 ID	主語を修飾する論理体の識別子を定義する。
10	定義体 ID	主語を修飾する定義体の識別子を定義する。
11	定義体種別	主語を修飾する定義体の種別を定義する。 例: 画面、帳票、RAWファイル、SEQファイル、電文、内部 TBL 等
12	定義体項目名	主語の定義体上における識別子を定義する。
13	Vector 種別	ベクトルの種別 (表 4) を定義する。
14		W02 W02 に属すベクトルの第 2 規約の作用を定義する。
15	第 2 規約	W03 未使用
16		W04 W04 に属すベクトルの第 2 規約の作用を定義する。
17		W02 未使用
18	第 3 規約	W03 W03 に属すベクトルの第 3 規約の作用を定義する。
19		W04 未使用
20	WORK 領域定義	ベクトルで用いられる補助領域を定義する。
21	READ 種別	主語が所属する論理体を取得する為の入力コマンドの種別を定義する。 例: READ、SELECT、FETCH 等
22	WRITE 種別	主語が所属する論理体を出力する為の出力コマンドの種別を定義する。 例: WRITE、INSERT、UPDATE、DELETE 等
23	型	主語の文字タイプを定義する。
24	桁数	主語の桁数を定義する。
25	小数桁数	主語の小数桁を定義する。
26	配列	主語の配列 (表 3) を定義する。

【図 3 4】

\* 本表の経路発報記録以外の全情報は、PRD情報から自動生成する。

項番	自SF (i, j)	自W03-ID	R種別	次 ΦP 数	隣々上位		隣上位		隣下位		経路発報 記録	作用R3C情報への 経路発報記録 (n個有)
					ΦP-ID	ΦP-ID	ΦP-ID	ΦP-ID	ΦP-ID	ΦP-ID		
1	1, 1	W03-1, 1	R3R	1	W04-1. 1	—	—	—	—	—		
2	2, 1	W03-2, 1	R3R	1	W04-2. 1	—	—	—	—	—		
			R3D	1	—	—	W03-1. 1	—	—	—		
			R3C	1	—	—	—	—	W04-3. 1	—		
3	3, 1	W03-3. 1	R3R	1	W04-3. 1	—	—	—	—	—		
			R3D	1	—	—	W03-2. 1	—	—	—		
			R3M	1	—	W04-1. 1	—	—	—	—		SF (2, 1) R3C

\* SF (i, j) は、PRD (図 4 参照) におけるSFの座標。

\* ΦP-IDは、パレット関数の識別子。

\* 経路発報記録は、パレット連鎖関数に組込まれる専用のプログラムで制御される。

\* 本表は、図 17 のPRDの例示である。R3Cは複数派生する事がある。

【図 35】

	主語				ベクトル 種別	ベクトルの管理項目							定義 規則 番号	備考
	種別	属性の組合せ				管理 番号	言語 区分	言語 Ver.	主語 種別 番号	ベクトル 種別	属性 の組合せ	集計 区分		
		配列 (列)	等価	境界										
1	正規				L2	0010	VB	66	01	L2	01	0	1	
2	正規				L3	0020	VB	66	01	L3	03	0	3	
3	正規				L4	0030	VB	66	01	L4	03	0	3	
4	正規	○			L2	0040	VB	66	01	L2	02	0	2	
5	正規	○			L3	0050	VB	66	01	L3	05	0	5	
6	正規	○			L4	0060	VB	66	01	L4	05	0	5	
7	正規		○		L3	0070	VB	66	01	L3	04	0	4	
8	正規		○		L4	0080	VB	66	01	L4	04	0	4	
9	正規	○	○		L3	0090	VB	66	01	L3	06	0	6	
10	正規	○	○		L4	0100	VB	66	01	L4	06	0	6	
11	正規			○	L3	0110	VB	66	01	L3	07	0	7	
12	正規			○	L4	0120	VB	66	01	L4	07	0	7	
13	正規		○	○	L3	0130	VB	66	01	L3	08	0	8	
14	正規		○	○	L4	0140	VB	66	01	L4	08	0	8	
15	正規	○		○	L3	0150	VB	66	01	L3	09	0	9	
16	正規	○		○	L4	0160	VB	66	01	L4	09	0	9	
17	正規	○	○	○	L3	0170	VB	66	01	L3	10	0	10	
18	正規	○	○	○	L4	0180	VB	66	01	L4	10	0	10	
19	K				L3	0190	VB	66	02	L3	11	0	11	
20	K				L4	0200	VB	66	02	L4	11	0	11	
21	K		○		L3	0210	VB	66	02	L3	12	0	12	
22	K		○		L4	0220	VB	66	02	L4	12	0	12	
23	K	○			L3	0230	VB	66	02	L3	13	0	13	
24	K	○			L4	0240	VB	66	02	L4	13	0	13	
25	K	○	○		L3	0250	VB	66	02	L3	14	0	14	
26	K	○	○		L4	0260	VB	66	02	L4	14	0	14	
27	M				L3	0290	VB	66	03	L3	17	0	15	
28	M				L4	0300	VB	66	03	L4	17	0	15	
29	M		○		L3	0310	VB	66	03	L3	18	0	16	
30	M		○		L4	0320	VB	66	03	L4	18	0	16	
31	M	○			L3	0330	VB	66	03	L3	19	0	17	
32	M	○			L4	0340	VB	66	03	L4	19	0	17	
33	M	○	○		L3	0350	VB	66	03	L3	20	0	18	
34	M	○	○		L4	0360	VB	66	03	L4	20	0	18	
35	入力アクセサ				L3	0370	VB	66	04	L3	22	0	20	
36	入力処理条件				L3	0390	VB	66	05	L3	22	0	21	
37	出力アクセサ				L3	0410	VB	66	07	L3	24	0	23	
38	出力処理条件				L3	0430	VB	66	08	L3	24	0	24	
39	入力コマンド				I2	0450	VB	66	06	I2	21	0	19	
40	出力コマンド				O4	0460	VB	66	09	O4	23	0	22	
41	経路				R4	0470	VB	66	10	R4	25	0	25	
42	経路				R2C	0480	VB	66	11	R2C	25	0	26	
43	経路				R2	0490	VB	66	11	R2	25	0	27	
44	経路				R3R	0500	VB	66	12	R3R	25	0	28	
45	正規				L3	0510	VB	66	01	L3	03	1	11	Kの派生元
46	正規				L4	0520	VB	66	01	L4	03	1	11	Kの派生元
47	正規	○			L3	0530	VB	66	01	L3	05	1	13	Kの派生元
48	正規	○			L4	0540	VB	66	01	L4	05	1	13	Kの派生元
49	正規		○		L3	0550	VB	66	01	L3	04	1	12	Kの派生元
50	正規		○		L4	0560	VB	66	01	L4	04	1	12	Kの派生元
51	正規	○	○		L3	0570	VB	66	01	L3	06	1	14	Kの派生元

【図 36】

	主語				ベクトル 種別	ベクトルの管理項目							定義 規則 番号	備考
	種別	属性の組合せ				管理 番号	言語 区分	言語 Ver.	主語 種別 番号	ベクトル 種別	属性の組合せ	集計 区分		
		配列 (列)	等価	境界										
52	正規	○	○		L4	0580	VB	66	01	L4	06	1	14	Kの派生元
53	正規			○	L3	0590	VB	66	01	L3	07	1	11&7	Kの派生元
54	正規			○	L4	0600	VB	66	01	L4	07	1	11&7	Kの派生元
55	正規		○	○	L3	0610	VB	66	01	L3	08	1	12&8	Kの派生元
56	正規		○	○	L4	0620	VB	66	01	L4	08	1	12&8	Kの派生元
57	正規	○		○	L3	0630	VB	66	01	L3	09	1	13&9	Kの派生元
58	正規	○		○	L4	0640	VB	66	01	L4	09	1	13&9	Kの派生元
59	正規	○	○	○	L3	0650	VB	66	01	L3	10	1	14&10	Kの派生元
60	正規	○	○	○	L4	0660	VB	66	01	L4	10	1	14&10	Kの派生元
61	K				L2	0670	VB	66	02	L2	91	0	11	領域のみ
62	K	○			L2	0680	VB	66	02	L2	92	0	13	領域のみ
63	S4 (12第2)				S4	0690	VB	66	13	S4	21	0	29	
64	S4 (12第4)				S4	0700	VB	66	14	S4	21	0	30	
65	S4 (04第4)				S4	0710	VB	66	15	S4	23	0	33	
66	S4 (L4第4)				S4	0720	VB	66	16	S4	03	0	36	
67	S4 (L4第4)		○		S4	0730	VB	66	16	S4	04	0	37	
68	S4 (L4第4)	○			S4	0740	VB	66	16	S4	05	0	38	
69	S4 (L4第4)	○	○		S4	0750	VB	66	16	S4	06	0	39	
70	S4 (L4第4)			○	S4	0760	VB	66	16	S4	07	0	40	
71	S4 (L4第4)		○	○	S4	0770	VB	66	16	S4	08	0	41	
72	S4 (L4第4)	○		○	S4	0780	VB	66	16	S4	09	0	42	
73	S4 (L4第4)	○	○	○	S4	0790	VB	66	16	S4	10	0	43	
74	S4 (L4第4)				S4	0800	VB	66	16	S4	11	0	44	
75	S4 (L4第4)		○		S4	0810	VB	66	16	S4	12	0	45	
76	S4 (L4第4)	○			S4	0820	VB	66	16	S4	13	0	46	
77	S4 (L4第4)	○	○		S4	0830	VB	66	16	S4	14	0	47	
78	S4 (L2第4)				S4	0880	VB	66	17	S4	01	0	48&50&51	
79	S4 (L2第4)	○			S4	0890	VB	66	17	S4	02	0	49&52&53	
80	S4 (L3第4)				S4	0900	VB	66	18	S4	03	0	54	
81	S4 (L3第4)		○		S4	0910	VB	66	18	S4	04	0	55	
82	S4 (L3第4)	○			S4	0920	VB	66	18	S4	05	0	56	
83	S4 (L3第4)	○	○		S4	0930	VB	66	18	S4	06	0	57	
84	S4 (L3第4)			○	S4	0940	VB	66	18	S4	07	0	58	
85	S4 (L3第4)		○	○	S4	0950	VB	66	18	S4	08	0	59	
86	S4 (L3第4)	○		○	S4	0960	VB	66	18	S4	09	0	60	
87	S4 (L3第4)	○	○		S4	0970	VB	66	18	S4	10	0	61	
88	S4 (L3第4)				S4	0980	VB	66	18	S4	11	0	62	
89	S4 (L3第4)		○		S4	0990	VB	66	18	S4	12	0	63	
90	S4 (L3第4)	○			S4	1000	VB	66	18	S4	13	0	64	
91	S4 (L3第4)	○	○		S4	1010	VB	66	18	S4	14	0	65	
92	S4 (L3第4)				S4	1020	VB	66	18	S4	17	0	66	
93	S4 (L3第4)		○		S4	1030	VB	66	18	S4	18	0	67	
94	S4 (L3第4)	○			S4	1040	VB	66	18	S4	19	0	68	
95	S4 (L3第4)	○	○		S4	1050	VB	66	18	S4	20	0	69	
96	S4 (L3第4)				S4	1060	VB	66	18	S4	22	0	31&32	
97	S4 (L3第4)				S4	1070	VB	66	18	S4	24	0	34&35	
98	正規				L4	1080	VB	66	01	L4	03	0	3	
99	正規	○			L4	1090	VB	66	01	L4	05	0	5	
100	正規		○		L4	1100	VB	66	01	L4	04	0	4	
101	正規	○	○		L4	1110	VB	66	01	L4	06	0	6	
102	正規			○	L4	1120	VB	66	01	L4	07	0	7	

【図 37】

	主語					ベクトル 種別	ベクトルの管理項目							定義 規則 番号	備考
	種別	属性の組合せ			管理 番号		言語 区分	言語 Ver.	主語 種別 番号	ベクトル 種別	属性の 組合せ	集計 区分			
		配列 (列)	等価	境界											
103	正規		○	○	L4	1130	VB	66	01	L4	08	0	8		
104	正規	○		○	L4	1140	VB	66	01	L4	09	0	9		
105	正規	○	○	○	L4	1150	VB	66	01	L4	10	0	10		
106	K				L4	1160	VB	66	02	L4	11	0	11		
107	K		○		L4	1170	VB	66	02	L4	12	0	12		
108	K	○			L4	1180	VB	66	02	L4	13	0	13		
109	K	○	○		L4	1190	VB	66	02	L4	14	0	14		
110	M				L4	1200	VB	66	03	L4	17	0	15		
111	M		○		L4	1210	VB	66	03	L4	18	0	16		
112	M	○			L4	1220	VB	66	03	L4	19	0	17		
113	M	○	○		L4	1230	VB	66	03	L4	20	0	18		
114	正規				L4	1280	VB	66	01	L4	03	1	11	Kの派生元	
115	正規	○			L4	1290	VB	66	01	L4	05	1	13	Kの派生元	
116	正規		○		L4	1300	VB	66	01	L4	04	1	12	Kの派生元	
117	正規	○	○		L4	1310	VB	66	01	L4	06	1	14	Kの派生元	
118	正規			○	L4	1320	VB	66	01	L4	07	1	11&7	Kの派生元	
119	正規		○	○	L4	1330	VB	66	01	L4	08	1	12&8	Kの派生元	
120	正規	○		○	L4	1340	VB	66	01	L4	09	1	13&9	Kの派生元	
121	正規	○	○	○	L4	1350	VB	66	01	L4	10	1	14&10	Kの派生元	
122	ダミー					9999	VB	66	99	99	99	0			
123	Φ0				Φ	2000	VB	66		W0			無		
124	Φ2				Φ2	2100	VB	66		W02			無		
125	Φ3				Φ3	2200	VB	66		W03			無		
126	Φ4				Φ4	2300	VB	66		W04			無		

【図 38】

型内の変数と L y e e B E L T 項目の対応を示す。

項番	項目名	項目 I D	トークン
1	名称	LyeeBELT_SubjectName	0x010
2	識別子	LyeeBELT_SubjectID	0x020
3	入出力区分	LyeeBELT_InOrOut	0x030
4	主語種別	LyeeBELT_SubjectKind	0x040
5	属性組み合わせの型	LyeeBELT_SubjectType	0x050
6	等価種別数	LyeeBELT_EquivalenceNo	0x060
7	等価識別子	LyeeBELT_EquivalenceID	0x070
8	主語座標	LyeeBELT_SelfCoordinate	0x080 (0x020, 0x280, 0x290, 0x300, 0x310) 内訳 0x020: 識別子, 0x280: PRD, 0x290: SF, 0x300: Φp, 0x310: 論理体
9	始点座標	LyeeBELT_ValCoordinate	0x090 (0x2000, 0x2850, 0x2950, 0x3050, 0x3150) 内訳 0x2000: 識別子, 0x2850: PRD, 0x2950: SF, 0x3050: Φp, 0x3150: 論理体
10	論理体 I D	LyeeBELT_LogicalDefID	0x100
11	定義体 I D	LyeeBELT_PhysicalDefID	0x100
12	定義体種別	LyeeBELT_MediumKind	0x120
13	定義体項目名	LyeeBELT_ObjectName	0x130
14	V e c t o r 種別	LyeeBELT_Subject_ID1	0x140
15	W 0 2 第 2 規則	LyeeBELT_W02BOX2	0x150
16	W 0 2 第 2 ( S U B )	LyeeBELT_W02BOX2WS	0x1510
17	W 0 2 第 2 ( W O R K )	LyeeBELT_W02BOX2WL	0x1520
18	W 0 3 第 2 規則	LyeeBELT_W02BOX4	0x1530
19	W 0 3 第 2 ( S U B )	LyeeBELT_W02BOX4WS	0x1540
20	W 0 3 第 2 ( W O R K )	LyeeBELT_W02BOX4WL	0x1550
21	W 0 4 第 2 規則	LyeeBELT_W03BOX2	0x160
22	W 0 4 第 2 ( S U B )	LyeeBELT_W03BOX2WS	0x1610
23	W 0 4 第 2 ( W O R K )	LyeeBELT_W03BOX2WL	0x1620
24	W 0 2 第 4 規則	LyeeBELT_W03BOX4	0x1630
25	W 0 2 第 4 ( S U B )	LyeeBELT_W03BOX4WS	0x1640
26	W 0 2 第 4 ( W O R K )	LyeeBELT_W03BOX4WL	0x1650
27	W 0 3 第 4 規則	LyeeBELT_W04BOX2	0x170
28	W 0 3 第 4 ( S U B )	LyeeBELT_W04BOX2WS	0x1710
29	W 0 3 第 4 ( W O R K )	LyeeBELT_W04BOX2WL	0x1720
30	W 0 4 第 4 規則	LyeeBELT_W04BOX4	0x180 境界座標/L2K座標の場合、追加 0x020, 0x2800, 0x2900, 0x3000, 0x3100 内訳 0x020: 識別子, 0x2800: PRD, 0x2900: SF, 0x3000: Φp, 0x3100: 論理体
31	W 0 4 第 4 ( S U B )	LyeeBELT_W04BOX4WS	0x1810
32	W 0 4 第 4 ( W O R K )	LyeeBELT_W04BOX4WL	0x1820
33	R E A D 種別	LyeeBELT_I2type	0x190
34	W R I T E 種別	LyeeBELT_O4type	0x210
35	型	LyeeBELT_DataType	0x230 (主語のデータ型) 0x2350 (始点単語のデータ型)
36	桁数	LyeeBELT_DataLength	0x240
37	小数桁数	LyeeBELT_Decimal	0x250
38	配列 (行)	LyeeBELT_RowNo	0x260 (主語の配列: 行) 0x2650 (始点単語の配列: 行)
39	配列 (列)	LyeeBELT_ColNo	0x270 (主語の配列: 列) 0x2750 (始点単語の配列: 列)
40	配列行順序列	LyeeBELT_ArrayRowOrder	0x470
41	配列列順序列	LyeeBELT_ArrayColOrder	0x480
42	集計区分	LyeeBELT_SumWord	0x490

## 【図 39】

\* プログラム言語がVBの場合の定義規則 1 の例

```

Option Explicit
Rem ##### 見出部 #####
Rem ***ユーザ名:
Rem ***システム名:
Rem ***定義規則: 1
Rem ***Vector主キー項目 *****
Rem ***管理番号: 0010
Rem ***編集指示区分: 0
Rem ***言語種別: VB
Rem ***言語バージョン: 66
Rem ***主語種別: 01
Rem ***ベクトル種別: L2
Rem ***属性の組合せ: 01
Rem ***集計: 0
Rem #主語名: @%01@
Rem #主語ID: @%02@
Rem ***座標 (PRD識別子_SF識別子_論理体識別子) : @%28@_@%29@_@%10@
Rem ***ベクトルのステータス*****
Rem ***ステータス情報:  @%STS@ ("0":完了," ":未完)
Rem ***生成年月日時分秒: @%TIME@
Rem ##### 領域部 #####
Rem #DEFPUB ** 公的領域部 -----
Rem PUB4N2 -----第4規約正規領域
  Private W@%30@_@%31@_@%28@_@%29@_@%02@ As @%23@
Rem PUBIN2 -----第2規約始点領域
  Private @%31@_@%28@_@%29@_@%02@ As @%23@
Rem PUB5C2 -----第5規約正規領域 状態変化フラグ (今回)
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y2 -----第5規約正規領域 状態変化フラグ (前回)
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B2 -----第5規約正規領域 状態変化フラグ (前々回)
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F2 -----第6規約正規領域 不成立フラグ
  Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV ** 私的領域部 -----
Rem PRV2T2 -----第2規約端点領域
  Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk As @%23@
Rem PRV7R2 -----第7規約再起領域
  Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU ----- 空の値
  Private CNS_NOT_KUH_@%23@ As @%23@
Rem $PRV2W2 -----第2規約ベクトル内作業領域 開始
@%152@
Rem $PRV2E2 -----第2規約ベクトル内作業領域 終了
Rem $PRV4W2 -----第4規約ベクトル内作業領域 開始
@%155@
Rem $PRV4E2 -----第4規約ベクトル内作業領域 終了
Rem #DEFEND
Rem ##### 論理部 #####
Rem # 主語ID: @%02@ 座標 (PRD_SF_論理体) : @%28@_@%29@_@%10@
Rem *-----
Public Sub Main0

```

## 【図 40】

\* プログラム言語がVBの場合の定義規則1の例

```
Rem #VECREP Private Sub L2_0%310_0%280_0%290_0%020 0
BOX_1:
Rem PRVLG
  If 0%300_0%310_0%280_0%290_0%020 = CNS_NOT_KUH_0%230 Then
    GoTo BOX_2
  End If
  GoTo BOX_E
BOX_2:
Rem PRVLG
  0%300_0%310_0%280_0%290_0%020_wk = 0%310_0%280_0%290_0%020
BOX_3:
Rem PRVLG
  If 0%300_0%310_0%280_0%290_0%020_wk < CNS_NOT_KUH_0%230 Then
    GoTo BOX_4
  End If
  GoTo BOX_5
BOX_4:
Rem PRVLG
  0%300_0%310_0%280_0%290_0%020 = 0%300_0%310_0%280_0%290_0%020_wk
Rem PRVLG
  CTRL_0%300_0%280_0%290_STS_TRANSITION_FLG
    = CTRL_0%300_0%280_0%290_STS_TRANSITION_FLG + 1
  GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_0%300_0%280_0%290_STS_TRANSITION_FLG_P = CTRL_0%300_0%280_0%290_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_0%300_0%290_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```



## 【図 4 1】

主語が「マージン」の場合の例

```

Option Explicit
Rem $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ 見出部 $$$$$$$$$$$$$$$$$$$$$$
Rem ***ユーザ名: Ube
Rem ***システム名:
Rem ***定義規則: 7
Rem ***Vector主キー項目 *****
Rem ***管理番号: 0120
Rem ***編集指示区分: 0
Rem ***言語種別: VB
Rem ***言語バージョン: 66
Rem ***主語種別: 01
Rem ***ベクトル種別: L4
Rem ***属性の組合せ: 07
Rem ***集計: 0
Rem #主語名: マージン
Rem #主語ID: MAJIN
Rem ***座標 (PRD識別子_SF識別子_論理体識別子): PRD1_SF15_Asnd
Rem ***ベクトルのステータス*****
Rem ***ステータス情報: "0" ("0":完了," ":未完)
Rem ***生成年月日時分秒: 03/07/21 13:44:38
Rem $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ 領域部 $$$$$$$$$$$$$$$$$$$$$$
Rem #DEFPUB ** 公的領域部 -----
Rem PUB4N4 -----第4規約正規領域
Private W4_Asnd_PRD1_SF15_MAJIN As Integer
Rem PUBIN4 -----第2規約始点領域
Private W2_Arec_PRD1_SF15_TEIKA As Integer
Private W4_Asnd_PRD1_SF15_AMUNT As Integer
Rem PUB5C4 -----第5規約正規領域 状態変化フラグ (今回)
Private CTRL_W4_PRD1_SF15_STS_TRANSITION_FLG As Integer
Rem PUB5Y4 -----第5規約正規領域 状態変化フラグ (前回)
Private CTRL_W4_PRD1_SF15_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4 -----第5規約正規領域 状態変化フラグ (前々回)
Private CTRL_W4_PRD1_SF15_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4 -----第6規約正規領域 不成立フラグ
Private CTRL_W4_Asnd_PRD1_SF15_MAJIN_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem PUB4B4 -----第4規約境界領域
Private W4_Asnd_PRD1_SF11_AMUNT As Integer
Rem PUB4N3 -----W3の第4規約正規領域
Private W3_Asnd_PRD1_SF15_MAJIN As Integer
Rem PUB4N2 -----W2の第4規約正規領域
Private W2_Asnd_PRD1_SF15_MAJIN As Integer
Rem #DEFPRV ** 私的領域部 -----
Rem PRV2T4 -----第2規約端点領域
Private W4_Asnd_PRD1_SF15_MAJIN_wk As Integer
Rem PRV7R4 -----第7規約再起領域
Private CTRL_W4_SF15_ING_FLG As Boolean
Rem #PUBKU 空の値
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single

```

## 【図 4 2】

主語が「マージン」の場合の例

```

Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Rem $PRV2W4 -----第2規約ベクトル内作業領域 開始
'   ワーク領域はありません
Rem $PRV2E4 -----第2規約ベクトル内作業領域 終了
Rem $PRV4W4 -----第4規約ベクトル内作業領域 開始
'   ワーク領域はありません
Rem $PRV4E4 -----第4規約ベクトル内作業領域 終了
Rem #DEFEND
Rem $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ 論理部 $$$$$$$$$$$$$$$$$$$$$$$$$$
Rem # 主語ID: MAJIN 座標 (PRD_SF_論理体) : PRD1_SF15_Asnd
Public Sub Main 0
Rem #VECREP Private Sub L4_Asnd_PRD1_SF15_MAJIN_10
BOX_1:
    If W4_Asnd_PRD1_SF15_MAJIN = CNS_NOT_KUH_Integer And _
        W3_Asnd_PRD1_SF15_MAJIN = 1 Then GoTo BOX_2
    End If
    GoTo BOX_E
BOX_2:
Rem PUBIN
    If W2_Arec_PRD1_SF15_TEIKA = CNS_NOT_KUH_Integer Then GoTo BOX_3
    If W4_Asnd_PRD1_SF15_AMUNT = CNS_NOT_KUH_Integer Then GoTo BOX_3
    W4_Asnd_PRD1_SF15_MAJIN_wk
        = Val (W2_Arec_PRD1_SF15_TEIKA * W4_Asnd_PRD1_SF15_AMUNT)
BOX_3:
    If W4_Asnd_PRD1_SF15_MAJIN_wk <> CNS_NOT_KUH_Integer Then GoTo BOX_4
    End If
    GoTo BOX_5
BOX_4:
    W4_Asnd_PRD1_SF15_MAJIN = W4_Asnd_PRD1_SF15_MAJIN_wk
    W4_Asnd_PRD1_SF15_AMUNT = W4_Asnd_PRD1_SF15_MAJIN
    CTRL_W4_PRD1_SF15_STS_TRANSITION_FLG
        = CTRL_W4_PRD1_SF15_STS_TRANSITION_FLG + 1
    GoTo BOX_E
BOX_5:
    If CTRL_W4_PRD1_SF15_STS_TRANSITION_FLG_P
        = CTRL_W4_PRD1_SF15_STS_TRANSITION_FLG_PP Then
        GoTo BOX_6
    End If
    GoTo BOX_7
BOX_6:
    CTRL_W4_Asnd_PRD1_SF15_MAJIN_NOT_VALID_FLG = True
    GoTo BOX_E
BOX_7:
    CTRL_W4_SF15_ING_FLG = True
    GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND

```

## 【図 43】

\* W02パレット関数の型 (1/3)

```

Option Explicit
Rem ***** 見出部 *****
Rem ***ユーザ名:
Rem ***システム名:
Rem ***パレット関数 *****
Rem ***管理番号:
Rem ***言語種別: VB
Rem ***言語バージョン: 66
Rem ***パレット種別: W02
Rem ***パレット識別子: W0%30@_@%28@_@%29@_@%02@
Rem ***製造ステータス情報: @%STS@ ('0':完了,' ':未完)
Rem ***生成年月日時分秒: @%TIME@
Rem ***** 領域部 *****
Rem #DEFPUB ** -----パレット連鎖関数領域 開始-@%28@_@%29@_@%02@-----
Rem *-----新次パレット指示領域
Private CTR_NEXT_PALLET_ID_NEW As String
Rem PUB4N2 -----W02 第4規約領域 (入力論理体の編集)
Rem ***-----W02 状態変化領域の編集****
Rem PUB5C2 -----第5規約 今回状態変化領域の編集
Rem PUB5Y2 -----第5規約 前回状態変化領域の編集
Rem PUB5B2 -----第5規約 前々回状態変化領域の編集
Rem PUB6F2 -----第6規約領域 不成立フラグの編集
Rem -----12今回状態変化領域の編集
Rem #DEFEND ** -----パレット連鎖関数領域 終り
Rem PUBIN2 -----W02 第2規約始点 領域
Rem *-----行インデックス
Public Gyou as Integer
Rem *-----列インデックス
Public Retu as Integer
Rem *-----内側のカウンタ
Public Uti_Cnt as Integer
Rem *-----外側のカウンタ
Public Soto_Cnt as Integer
Rem *-----内側の最大値
Public Uti_Max as Integer
Rem *-----外側の最大値
Public Soto_Max as Integer
Rem #DEFFIN
Rem #DEFPRV ** -----W02パレット関数領域 開始-----@%28@_@%29@_@%02@-----
Rem PRV2T2 -----第2規約端点領域の編集
Rem PRV7R2 -----第7規約再起フラグ領域の編集
Rem *****ベクトル内ワーク領域の編集-----*****
Rem PRV2W2 -----第2規約ベクトル内作業領域
Rem PRV4W2 -----第4規約ベクトル内作業領域
Rem SUBの領域編集 (サブルーチン単位)
Rem 引渡領域
Rem 受取領域
Rem *-----PUBKU 空の値
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single

```

## 【図 4 4】

\*W02パレット関数の型 (2/3)

```

Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Rem #DEFEND ** W02パレット関数領域 終り-----
'
' *****
' * W02 パレット関数 (Φ2) *
' *****
Rem PalID
Public Sub @%PALID@()
MAIN_START:
' -----< 状態Fクリア >-----'
Rem PRVLG
CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = 0
Rem PRVLG
CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P = 0
Rem PRVLG
CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP = 0
Rem PRVLG
CTRL_W@%30@_@%28@_@%29@_I2_STS_TRANSITION_FLG = 0
' -----< L2第2領域のクリア >-----'
Rem PAL2T2
' -----< L2第6領域のクリア >-----'
Rem PAL6F2
' -----< L2第7領域のクリア >-----'
Rem PAL7F2
Rem *VecID -----< 入力作用要素I2 の配置命令 >-----'
Rem CALLVA
' -----< 再起からの戻り地点 >-----'
BOX_@%PALID@_RERUN_POINT:
' -----< 状態変化Fの処理 >-----'
' 前回→前々回
Rem PRVLG
CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP _
Rem PRVLG
= CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
' 今回→前回
Rem PRVLG
CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P _
Rem PRVLG
= CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG
Rem *VecID -----< 論理要素 L2 の配置命令 >-----'
Rem CALLVL
' -----< 再起判定 (前回<>前々回) >-----'
Rem PRVLG
If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P <> _
Rem PRVLG
CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
GoTo BOX_@%PALID@_RERUN_POINT End If
Rem *VecID -----< 経路作用要素R2Cの配置命令 >-----'
Rem CALLVR
Rem *VecID -----< 経路作用要素R2 の配置命令 >-----'

```

## 【図 45】

\*W02パレット関数の型(3/3)

```
Rem CALLVR
MAIN_END:
End Sub
' -----< W02パレット関数終了 >-----'
Rem * ##### {Set of Vector} #####
Rem VECPRC ***** I2, L2, R2C, R2の配置
Rem PRCEND
Rem * #####W02パレットおわり#####
```

## 【図 46】

\*W03パレット関数の型(1/2)

```

Option Explicit
Rem ***** 見出部 *****
Rem ***ユーザ名:
Rem ***システム名:
Rem ***パレット関数 *****
Rem ***管理番号:
Rem ***言語種別: VB
Rem ***言語バージョン: 66
Rem ***パレット種別: W03
Rem ***パレット識別子: W0306_0%286_6%290_6%020
Rem ***製造ステータス情報: 0%STS0 ('0':完了,' ':未完)
Rem ***生成年月日時分秒: 0%TIME0
Rem ***** 領域部 *****
Rem #DEFPPUB ** ----- パレット連鎖関数領域 開始-0%280_0%290_0%020-----
Rem *-----新次パレット指示領域
Private CTR_NEXT_PALLET_ID_NEW As String
Rem PUB4N3 -----W03 第4規約領域 D24
Rem *****W03 状態変化領域の編集*****
Rem PUB5C3 -----第5規約領域 今回状態変化領域の編集
Rem PUB5Y3 -----第5規約領域 前回状態変化領域の編集
Rem PUB5B3 -----第5規約領域 前々回状態変化領域の編集
Rem PUB6F3 -----第6規約領域 D26不成立フラグの編集
Rem #DEFEND ** -----パレット連鎖関数領域 終り-----
Rem PUB1N3 -----W03 第2規約始点 領域
Rem *-----行インデックス
Public Gyou as Integer
Rem *-----列インデックス
Public Retu as Integer
Rem *-----内側のカウンタ
Public Uti_Cnt as Integer
Rem *-----外側のカウンタ
Public Soto_Cnt as Integer
Rem *-----内側の最大値
Public Uti_Max as Integer
Rem *-----外側の最大値
Public Soto_Max as Integer
Rem #DEFPRV ** -----W03パレット関数領域 開始-----0%280_0%290_0%020-----
Rem PRV2T3 -----第2規約端点領域の編集
Rem PRV7R3 -----第7規約再起フラグ領域の編集
Rem *****ベクトル内ワーク領域の編集*****
Rem PRV2W3 -----第2規約ベクトル内作業領域
Rem PRV4W3 -----第4規約ベクトル内作業領域
Rem SUBの領域編集(サブルーチン単位)
Rem 引渡領域
Rem 受取領域
Rem *-----PUBKU 空の値
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant

```

## 【図 47】

\* W03 バレット関数の型 (2/2)

```
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Rem #DEFEND ** W03バレット関数領域 終り-----
'
' *****
' * W03 バレット関数 (Φ3) *
' *****
Rem PaLID
Public Sub @%PALID@()
MAIN_START:
' -----< 状態変化Fクリア >-----'
Rem PRVLG
CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = 0
Rem PRVLG
CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P = 0
Rem PRVLG
CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP = 0
' -----< L3第2領域のクリア >-----'
Rem PAL2T3
' -----< L3第6領域のクリア >-----'
Rem PAL6F3
' -----< L3第7領域のクリア >-----'
Rem PAL7F3
' -----< 再起からの戻り地点 >-----'
BOX_@%PALID@_RERUN_POINT:
' -----< 状態変化Fの処理 >-----'
' 前回→前々回
Rem PRVLG
CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP _
Rem PRVLG
= CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
' 今回→前回
Rem PRVLG
CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P _
Rem PRVLG
= CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG
' -----< L3の配置命令 >-----'
Rem CALLVL
' -----< 再起判定 (前回<>前々回) >-----'
Rem PRVLG
If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P <> _
Rem PRVLG
CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
GoTo BOX_@%PALID@_RERUN_POINT End If
Rem *VecID -----< 経路作用要素R3R配置命令 >-----'
Rem CALLVR
MAIN_END:
End Sub
' -----< W03バレット関数終了 >-----'
Rem * ##### (Set of Vector) #####
Rem VECPRC ***** L3, R3Rの配置
Rem PRCEND
Rem * #####W03バレットおわり#####
```

## 【図 48】

\*W04パレット関数の型 (1/3)

```

Option Explicit
Rem ***** 見出部 *****
Rem ***ユーザ名:
Rem ***システム名:
Rem ***パレット関数 *****
Rem ***管理番号:
Rem ***言語種別: VB
Rem ***言語バージョン: 66
Rem ***パレット種別: W04
Rem ***パレット識別子: W04300_0280_0290_0200
Rem ***製造ステータス情報: 0STS0 ('0':完了,' ':未完)
Rem ***生成年月日時分秒: 0TIME0
Rem ***** 領域部 *****
Rem #DEFPUB ** -----パレット連鎖関数領域 開始-0280_0290_0200-----
Rem *-----新次パレット指示領域
Private CTR_NEXT_PALLET_ID_NEW As String
Rem PUB4N4 -----W04 第4規約領域 (出力論理体の編集)
Rem PUB1N4 -----W04 第2規約始点 領域
Rem PUB4B4 -----W04 規約境界領域
Rem ****-----W04 状態変化領域の編集****
Rem PUB5C4 -----第5規約領域 今回状態変化領域の編集
Rem PUB5Y4 -----第5規約領域 前回状態変化領域の編集
Rem PUB5B4 -----第5規約領域 前々回状態変化領域の編集
Rem PUB6F4 -----第6規約領域 不成立フラグの編集
Rem -----04今回状態変化領域の編集
Rem #DEFEND ** -----パレット連鎖関数領域 終り-----
Rem PUB4N3 -----W04 第1規約領域 (W03 第4規約領域)
' *****配列操作ワーク領域*****
Rem *-----行インデックス
Public Gyou as Integer
Rem *-----列インデックス
Public Retu as Integer
Rem *-----内側のカウンタ
Public Uti_Cnt as Integer
Rem *-----外側のカウンタ
Public Soto_Cnt as Integer
Rem *-----内側の最大値
Public Uti_Max as Integer
Rem *-----外側の最大値
Public Soto_Max as Integer
Rem #DEFFIN
' *****
Rem #DEFPRV ** -----W04パレット関数領域 開始-----0280_0290_0200-----
Rem PRV2T4 -----第2規約端点領域の編集
Rem PRV7R4 -----第7規約再起フラグ領域の編集
Rem ****-----ベクトル内ワーク領域の編集****
Rem PRV2W4 -----第2規約ベクトル内作業領域
Rem PRV4W4 -----第4規約ベクトル内作業領域
Rem SUBの領域編集 (サブルーチン単位)
Rem 引渡領域
Rem 受取領域
Rem *-----PUBKU 空の値
Private CNS_NOT_KUH_String As String

```



## 【図 49】

\*W04パレット関数の型 (2/3)

```

Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Rem #DEFEND ** W04パレット関数領域 終り-----
Rem * #####
      ' *****
      ' * W04 パレット関数 (Φ4) *
      ' *****
Rem PalID
Public Sub @%PALID@()
MAIN_START:
      ' -----< 状態変化Fクリア >-----'
Rem PRVLG
      CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = 0
Rem PRVLG
      CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P = 0
Rem PRVLG
      CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP = 0
Rem PRVLG
      CTRL_W@%30@_@%28@_@%29@_04_STS_TRANSITION_FLG = 0
      ' -----< L4第2領域のクリア >-----'
Rem PAL2T4
      ' -----< ML4第4領域のクリア >-----'
Rem PAL4M4
      ' -----< L4第6領域のクリア >-----'
Rem PAL6F4
      ' -----< L4第7領域のクリア >-----'
Rem PAL7F4
      ' -----< 再起からの戻り地点 >-----'
BOX_@%PALID@_RERUN_POINT:
      ' -----< 状態変化Fの処理 >-----'
      ' 前回→前々回
Rem PRVLG
      CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP _
Rem PRVLG
      = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
      ' 今回→前回
Rem PRVLG
      CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P _
Rem PRVLG
      = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG
      ' -----< ベクトルの配置命令 >-----'
      ' -----< L4の配置命令 >-----'
Rem CALLVL
      ' -----< 再起判定 (前回<>前々回) >-----'
Rem PRVLG
      If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P < _

```

## 【図 50】

\*W04バレット関数の型(3/3)

```
Rem PRVLG
      CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG_PP Then
      GoTo BOX_@%PALID0_RERUN_POINT
    End If
Rem *VecID -----< 出力作用要素04 (WRITE関係処理) 配置命令 >-----'
Rem CALLVA
Rem *VecID -----< 経路作用要素R4配置命令 >-----'
Rem CALLVR
Rem *VecID -----< 同期作用要素S4配置命令 >-----'
Rem CALLVS
MAIN_END:
End Sub
      -----< W04バレット関数終了 >-----'
Rem * ##### {Set of Vector} #####
Rem VECPRC ***** L4, O4, R4, S4の配置

Rem PRCEND
Rem * #####W04バレットおわり#####
```

【図 5 1】

```

Option Explicit
Rem #####
Rem ***ユーザ名:
Rem ***システム名:
Rem ***パレット連鎖関数 ***** (PRD 1 個につき 1 本定義する)
Rem ***管理番号:
Rem ***言語種別: VB
Rem ***言語バージョン: 66
Rem ***パレット連鎖関数識別子: @%PRDID@_@%PRDNAME@
Rem ***製造ステータス情報: @%STS@ ('0':完了,' ':未完)
Rem ***生成年月日時分秒: @%TIME@
Rem #####
Rem ***** パレット連鎖関数領域 *****
'..... 経路情報制御テーブル関連領域 開始 .....
Type RouteControlTableType
    No As Integer          ' 連番
    SFID As String         ' PRD情報TBLのブロックID
    Row As Integer         ' PRD情報TBLのRow
    Col As Integer         ' PRD情報TBLのCol
    PalID As String        ' PRD情報TBLのW03のパレットID
    SubNo As Integer       ' SFID、パレットID内での連番
    RouteCode As String    ' R3C、R3D、R3Mのいずれか
    NextPallets As Variant ' PRD情報TBLの経路種別毎のパレットID数
    NPID_R3R As String     ' 経路種別R3Rの場合、PRD情報のR3RのパレットIDを設定
    NPID_R3M As String     ' 経路種別R3Mの場合、PRD情報のR3MのパレットIDを設定
    NPID_R3D As String     ' 経路種別R3Dの場合、PRD情報のR3DのパレットIDを設定
    NPID_R3C As String     ' 経路種別R3Cの場合、PRD情報のR3CのパレットIDを設定
    SendedRouteID As String ' 経路発報記録
    SendedR3C_W03ID(10) As String ' 作用R3C情報への経路発報記録 (W03パレットID)
    SendedR3C_W04ID(10) As String ' 作用R3C情報への経路発報記録 (W04パレットID)
End Type
Public RCT(10000) As RouteControlTableType
Private InputRCT As String
Private ForSplit0 As String
Private ForSplit20 As String
Private RctIndex As Integer
Private RctIndex1 As Integer
Private RctIndex2 As Integer
Private RctIndex3 As Integer
Private RctIndex4 As Integer
Private RctIndex5 As Integer
Private RctIndex6 As Integer
Private RctIndex7 As Integer
Private RctIndex8 As Integer
Private RctIndex9 As Integer
Private RctIndex10 As Integer
Private RctIndex11 As Integer
Private RctIndex12 As Integer
Private RctIndex13 As Integer
Private RctIndex14 As Integer
Private RctIndex15 As Integer
Private RctIndex16 As Integer
Private RctIndex17 As Integer
Private RctIndex18 As Integer

```

【図 52】

```

Private RctIndex19 As Integer
Private RctIndex20 As Integer
Private PalletID_Now As String
'..... 経路情報制御テーブル関連領域 終了'.....
'*****配列操作用ワーク領域*****
Rem *-----行インデックス
Public Gyou As Integer
Rem *-----列インデックス
Public Retu As Integer
Rem *-----内側のカウンタ
Public Uti_Cnt As Integer
Rem *-----外側のカウンタ
Public Soto_Cnt As Integer
Rem *-----内側の最大値
Public Uti_Max As Integer
Rem *-----外側の最大値
Public Soto_Max As Integer
Rem *-----新次パレット指示領域
Public CTR_NEXT_PALLET_ID_NEW As String
Rem *-----旧次パレット指示領域
Public CTR_NEXT_PALLET_ID_OLD As String
Rem 第4領域の編集
Rem 第6領域の編集
Rem 状態領域の編集
Rem 今回状態領域の編集
Rem 前回状態領域の編集
Rem 前々回状態領域の編集
Rem 今回12状態領域の編集
Rem 今回04状態領域の編集
Rem #PRDDEF_PUB *** PRD公的領域部 --PrdID = @%01@ PrdName = @%02@
Rem #PRDDEF_PRIV *** PRD共通領域部
Rem #DEFPUB ** SF別公的領域部 -----SfID = @%11@ SfName = @%12@
Rem DEFPUB ** SF別公的領域部 取り込み
Rem DEFEND
Rem パレット連鎖関数内ワーク領域の編集
Rem パレット連鎖関数SUBの領域編集 (サブルーチン単位)
Rem 引渡領域
Rem 受取領域
Rem *-----経路制御テーブル (PRD 1 個につき 1 つ定義する)
' (PRD 情報テーブルから取得する)
Rem #PUBKU 空の値
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
'
' *****
' * パレット連鎖関数 (Φ) *
' *****

```

【図 53】

```

Rem #0020
Rem *-----
Public Sub @FormID@_Form_Load()
    Call TenseInit
    CTR_NEXT_PALLET_ID_NEW = '@EventProcID02'
    Call T0
End Sub
Rem *-----
Public Sub @FormID@_EventProcID@(@Para@)
    W2_@論理体ID@_@PrdID@_@SfID@_@Para@ = @Para@
    CTR_NEXT_PALLET_ID_NEW = '@EventProcID02'
    Call T0
End Sub
Rem *----- Φ0 バレット連鎖関数 *****
Public Sub T0()
    On Error GoTo OnErrorBox
BOX_RETURN_A:
    Rem PUB404
    ' *---<システム 終了判定 >---'
Rem PUB612
Rem PUB604
Rem *-----経路処理
    If CTR_NEXT_PALLET_ID_NEW = "" Then
        Call SelectNextPallet
    Else
        CTR_NEXT_PALLET_ID_OLD = CTR_NEXT_PALLET_ID_NEW      '旧次バレット指示領域←新次バレット指示領域
        CTR_NEXT_PALLET_ID_NEW = ""                          '新次バレット指示領域の値をクリア
    End If
Rem *-----バレット起動処理（旧次バレット指示領域に従う）
    Call RunPallets
Rem *-----戻り地点Aに戻る
    Go To BOX_RETURN_A
Rem *-----終了処理
BOX_END:
    END
OnErrorBox:
    ' 運用条件に応じた特殊処理
    END
End Sub
Rem *** PRCEND
' ****初期処理*****
Sub TenseInit()
    ' -----< 経路制御テーブルの作成 ----->
    Call MakeRouteControlTable
    ' -----< バレット連鎖関数領域の初期化 ----->
Rem *-----バレット連鎖関数が管理している領域をクリア
    CNS_NOT_KUH_String = ChrW(0)      ' As String
    CNS_NOT_KUH_Integer = -32768      ' As Integer
    CNS_NOT_KUH_Boolean = False      ' As Boolean
    CNS_NOT_KUH_Long = -2147483648#   ' As Long
    CNS_NOT_KUH_Single = -3.402823E+38 ' As Single
    CNS_NOT_KUH_Double = -4.94065645841247E-324 ' As Double
    CNS_NOT_KUH_Variant = -4.94065645841247E-324 ' As Variant
    CNS_NOT_KUH_Currency = -922337203685477# ' As Currency

```

【図 5 4】

```

CNS_NOT_KUH_Byte = 255                'As Byte
CNS_NOT_KUH_Date = 999999             'As Date
Rem *-----初期起動パレットIDをセット
    Call SetFirstPalletId
End Sub
'****経路処理*****
Sub SelectNextPallet()
    RctIndex1 = 0 '自R3Rの位置指定ポインタ
    RctIndex2 = 0 '自R3C, R3D, R3Mの存在チェック用ポインタ
    RctIndex3 = 0 '自SFの座標=先頭 & 自SFの任意のR3C=発報済を指定するポインタ
    RctIndex4 = 0 '自R3Cの発報済数をカウント
    RctIndex5 = 0 '自PRDの全ての発報記録のクリア用カウンタ
    RctIndex6 = 0 '自R3D指示ポインタ
    RctIndex7 = 0 '自R3C指示ポインタ
    RctIndex8 = 0 '自R3M指示ポインタ
    RctIndex9 = 0 '自SFに作用する全てのR3Cを検索するポインタ
    RctIndex10 = 0 '自R3M指示ポインタ
    RctIndex11 = 0 '自R3C指示ポインタ
    RctIndex12 = 0 '自R3C指示ポインタ
    RctIndex13 = 0 '自R3C指示ポインタ
    RctIndex14 = 0 '自SFの全R3C発報済CHK用カウンタ
    RctIndex15 = 0 '経路制御情報テーブルの自R3D&自R3M発報記録のクリア
    RctIndex16 = 0 '未発報の自R3Cを発報用ポインタ
    RctIndex17 = 0 '他R3Cのレコードを指示するポインタ
    RctIndex18 = 0 'RctIndex18: 自R3Mのレコードの、作用R3C情報を指示するポインタ
    RctIndex19 = 0 '他R3Cのレコードを指示するポインタ
    RctIndex20 = 0

    '-----<現在実行中のパレットIDを取得>-----
    PalletID_Now = CTR_NEXT_PALLET_ID_OLD

    '-----<自R3R未発報?>-----

    Do Until RCT(RctIndex1).No < 1
        If Mid(RCT(RctIndex1).PalID, 1, Len(RCT(RctIndex1).PalID) - 1) _
            = Mid(PalletID_Now, 1, Len(PalletID_Now) - 1) _
            And RCT(RctIndex1).RouteCode = "R3R" Then
            '-----この時、自R3Rが有り-----
            If RCT(RctIndex1).SendedRouteID <> "1" Then
                '-----この時、自R3R未発報である-----
                '-----①発報「R3R」ΦP-IDを次ΦP-ID 指示エリアにセット-----
                CTR_NEXT_PALLET_ID_OLD = RCT(RctIndex1).NPID_R3R
                CTR_NEXT_PALLET_ID_NEW = ""
                '-----②経路制御TBLの自R3D発報記録の更新-----
                RCT(RctIndex1).SendedRouteID = "1"
                '-----経路選択処理終了-----
                RctIndex1 = 0
                RctIndex2 = 0
                RctIndex3 = 0
                RctIndex4 = 0
                RctIndex5 = 0
                RctIndex6 = 0
                RctIndex7 = 0
            End If
        End If
    Loop

```

【図 5 5】

```

        RctIndex8 = 0
        RctIndex9 = 0
        RctIndex10 = 0
        RctIndex11 = 0
        RctIndex12 = 0
        RctIndex13 = 0
        RctIndex14 = 0
        RctIndex15 = 0
        RctIndex16 = 0
        RctIndex17 = 0
        RctIndex18 = 0
        RctIndex19 = 0
        RctIndex20 = 0
        Exit Sub
    Else
        '-----この時、自R3R発報済である-----'
        GoTo BOX_IO_TRANSITION_CHK
    End If
End If
RctIndex1 = RctIndex1 + 1
Loop
' I/O状態変化あり?
BOX_IO_TRANSITION_CHK:
Rem PRVLG
' If PalletID_Now = @%100@ And (CTRL_W@%30@_@%28@_@%29@_12_STS_TRANSITION_FLG <> 0 Or
CTRL_W@%30@_@%28@_@%29@_04_STS_TRANSITION_FLG <> 0) then GoTo BOX_IO_TRANSITION_ON
        GoTo BOX_IO_TRANSITION_OFF ' 状態変化なしなら、BOX_IO_TRANSITION_OFFへ進む
BOX_IO_TRANSITION_ON:
    ' この時、I/O状態変化あり
    '-----①発報「R3R」ΦP-IDを次ΦP-ID 指示エリアにセット-----'
    CTR_NEXT_PALLET_ID_OLD = RCT(RctIndex1).NPID_R3R
    CTR_NEXT_PALLET_ID_NEW = ""
    '-----②経路制御TBLの自R3D発報記録の更新-----'
    RCT(RctIndex1).SendedRouteID = "I"
    '-----経路選択処理終了-----'
    RctIndex1 = 0
    RctIndex2 = 0
    RctIndex3 = 0
    RctIndex4 = 0
    RctIndex5 = 0
    RctIndex6 = 0
    RctIndex7 = 0
    RctIndex8 = 0
    RctIndex9 = 0
    RctIndex10 = 0
    RctIndex11 = 0
    RctIndex12 = 0
    RctIndex13 = 0
    RctIndex14 = 0
    RctIndex15 = 0
    RctIndex16 = 0
    RctIndex17 = 0
    RctIndex18 = 0
    RctIndex19 = 0

```

【図 5 6】

```

        RctIndex20 = 0
    Exit Sub
BOX_IO_TRANSITION_OFF:
    ' この時、I/O状態変化なし
    ' 自SFにR3C、R3D、R3Mのいずれかが存在するか?
    Do Until RCT(RctIndex2).No < 1
        If Mid(RCT(RctIndex2).PalID, 1, Len(RCT(RctIndex2).PalID) - 1) _
            = Mid(PalletID_Now, 1, Len(PalletID_Now) - 1) _
            And (RCT(RctIndex2).RouteCode = "R3C" _
                Or RCT(RctIndex2).RouteCode = "R3D" _
                Or RCT(RctIndex2).RouteCode = "R3M") Then
            ' -----この時、自R3M or 自R3C or 自R3Dが有り-----
            GoTo BOX_SELECT_CDM
        End If
        RctIndex2 = RctIndex2 + 1
    Loop
    ' -----この時、自R3M or 自R3C or 自R3Dが無し-----
    ' -----①発報「R3R」のP-IDを次のP-ID指示エリアにセット-----
    CTR_NEXT_PALLET_ID_OLD = RCT(RctIndex1).NPID_R3R
    CTR_NEXT_PALLET_ID_NEW = ""
    ' -----②経路制御TBLの自R3D発報記録の更新-----
    RCT(RctIndex1).SendedRouteID = "1"
    ' -----経路選択処理終了-----
    RctIndex1 = 0
    RctIndex2 = 0
    RctIndex3 = 0
    RctIndex4 = 0
    RctIndex5 = 0
    RctIndex6 = 0
    RctIndex7 = 0
    RctIndex8 = 0
    RctIndex9 = 0
    RctIndex10 = 0
    RctIndex11 = 0
    RctIndex12 = 0
    RctIndex13 = 0
    RctIndex14 = 0
    RctIndex15 = 0
    RctIndex16 = 0
    RctIndex17 = 0
    RctIndex18 = 0
    RctIndex19 = 0
    RctIndex20 = 0
    Exit Sub
BOX_SELECT_CDM: ' 自R3M or 自R3C or 自R3Dを選択する処理
    ' 自SFの座標=先頭 & 自SFの全R3C=発報済?
    ' RctIndex3: 自SFの座標=先頭 & 自SFの任意のR3C=発報済を指定するポインタ
    ' RctIndex4: R3Cの発報済数をカウント
    ' RctIndex5: 経路情報テーブルの、自PRDの全ての発報記録のクリアを制御するポイン
    Do Until RCT(RctIndex3).No < 1
        If Mid(RCT(RctIndex3).PalID, 1, Len(RCT(RctIndex3).PalID) - 1) _
            = Mid(PalletID_Now, 1, Len(PalletID_Now) - 1) _
            And RCT(RctIndex3).Col = 1 _
            And RCT(RctIndex3).Row = 1 _

```



【図 57】

```

And RCT(RctIndex3).RouteCode = "R3C" _
And RCT(RctIndex3).SendedRouteID = "1" Then
'-----この時、自SFの座標=先頭 & 自SFの任意のR3C=発報済-----
RctIndex4 = RctIndex4 + 1
If RctIndex4 = RCT(RctIndex3).NextPallels Then
'-----この時、自SFの座標=先頭 & 自SFの全R3C=発報済-----
' 経路情報テーブルの、自PRDの全ての発報記録のクリア
Do Until RCT(RctIndex5).No < 1
    RCT(RctIndex5).SendedRouteID = ""
    RctIndex5 = RctIndex5 + 1
Loop
Exit Do
End If
End If
RctIndex3 = RctIndex3 + 1
Loop
'-----
'-----<自R3Dが有り & 自R3D未発報?>-----
'-----
RctIndex6 : 自R3D指示ポインタ
RctIndex7 : 自R3C指示ポインタ
RctIndex8 : 自R3M指示ポインタ
Do Until RCT(RctIndex6).No < 1
    If Mid(RCT(RctIndex6).PalID, 1, Len(RCT(RctIndex6).PalID) - 1) _
        = Mid(PalletID_Now, 1, Len(PalletID_Now) - 1) _
    And RCT(RctIndex6).RouteCode = "R3D" _
    And RCT(RctIndex6).SendedRouteID <> "1" Then
'-----この時、自R3Dが有り & 自R3D未発報である-----
'-----①発報R3DのP-IDを次P-ID エリアにセット-----
CTR_NEXT_PALLET_ID_OLD = RCT(RctIndex6).NPID_R3D
CTR_NEXT_PALLET_ID_NEW = ""
'-----②経路制御TBLの自R3D発報記録の更新-----
RCT(RctIndex6).SendedRouteID = "1"
'-----経路制御TBLに自SFにR3Cが有るか?-----
RctIndex7 = 0
Do Until RCT(RctIndex7).No < 1
    If Mid(RCT(RctIndex7).PalID, 1, Len(RCT(RctIndex7).PalID) - 1) _
        = Mid(PalletID_Now, 1, Len(PalletID_Now) - 1) _
    And RCT(RctIndex7).RouteCode = "R3C" Then
'-----この時、自R3Cが有。パレット起動処理へ。-----
RctIndex1 = 0
RctIndex2 = 0
RctIndex3 = 0
RctIndex4 = 0
RctIndex5 = 0
RctIndex6 = 0
RctIndex7 = 0
RctIndex8 = 0
RctIndex9 = 0
RctIndex10 = 0
RctIndex11 = 0
RctIndex12 = 0
RctIndex13 = 0
RctIndex14 = 0

```

【図 58】

```

        RctIndex15 = 0
        RctIndex16 = 0
        RctIndex17 = 0
        RctIndex18 = 0
        RctIndex19 = 0
        RctIndex20 = 0
        Exit Sub
    End If
    RctIndex7 = RctIndex7 + 1
Loop
'この時、経路制御TBLに自SFにR3Cが存在しないから
'-----経路制御TBLに自SFにR3Mが有るか?-----'
    RctIndex8 = 0
    Do Until RCT(RctIndex8).No < 1
        If Mid(RCT(RctIndex8).PalID, 1, Len(RCT(RctIndex8).PalID) - 1) _
            = Mid(PalletID_Now, 1, Len(PalletID_Now) - 1) _
            And RCT(RctIndex8).RouteCode = "R3M" Then
            '-----この時、自R3Mが有。パレット起動処理へ。-----'
            RctIndex1 = 0
            RctIndex2 = 0
            RctIndex3 = 0
            RctIndex4 = 0
            RctIndex5 = 0
            RctIndex6 = 0
            RctIndex7 = 0
            RctIndex8 = 0
            RctIndex9 = 0
            RctIndex10 = 0
            RctIndex11 = 0
            RctIndex12 = 0
            RctIndex13 = 0
            RctIndex14 = 0
            RctIndex15 = 0
            RctIndex16 = 0
            RctIndex17 = 0
            RctIndex18 = 0
            RctIndex19 = 0
            RctIndex20 = 0
            Exit Sub
        End If
        RctIndex8 = RctIndex8 + 1
    Loop
    '-----この時、自R3Mが無-----'
    '-----経路制御TBLの自R3D発報記録のクリア-----'
    RCT(RctIndex6).SendedRouteID = ""
    '-----自SFのR3Dが作用する全てのR3Cの経路制御TBLの発報記録をオン-----'
    'RctIndex6:自R3Dのレコードを指示するポイント
    'RctIndex9:自R3Dのレコードの、作用R3C情報を指示するポイント
    'RctIndex17:他R3Cのレコードを指示するポイント
    RctIndex9 = 0
    Do Until RctIndex9 = 10
        If RCT(RctIndex6).SendedR3C_W03ID(RctIndex9) = "" Then Exit Do
        RctIndex17 = 0
        Do Until RCT(RctIndex17).No < 1

```

【図 59】

```

If RCT(RclIndex17).PalID = RCT(RclIndex6).SendedR3C_W03ID(RclIndex9) _
And RCT(RclIndex17).RouteCode = "R3C" _
And RCT(RclIndex17).NPID_R3C = RCT(RclIndex6).SendedR3C_W04ID(RclIndex9) Then
    RCT(RclIndex17).SendedRouteID = "I"
End If
RclIndex17 = RclIndex17 + 1
Loop
RclIndex9 = RclIndex9 + 1
Loop
'*****
'-----パレット起動処理へ。-----
RclIndex1 = 0
RclIndex2 = 0
RclIndex3 = 0
RclIndex4 = 0
RclIndex5 = 0
RclIndex6 = 0
RclIndex7 = 0
RclIndex8 = 0
RclIndex9 = 0
RclIndex10 = 0
RclIndex11 = 0
RclIndex12 = 0
RclIndex13 = 0
RclIndex14 = 0
RclIndex15 = 0
RclIndex16 = 0
RclIndex17 = 0
RclIndex18 = 0
RclIndex19 = 0
RclIndex20 = 0
Exit Sub
End If
RclIndex6 = RclIndex6 + 1
Loop
'このとき、R3Dの発報条件ではないから、自R3Mが有り&自R3M未発報済?の処理へ進む。
'-----<自R3Mが有り&自R3M未発報?>-----
'RclIndex10: 自R3M指示ポインタ
'RclIndex11: 自R3C指示ポインタ
Do Until RCT(RclIndex10).No < 1
    If Mid(RCT(RclIndex10).PalID, 1, Len(RCT(RclIndex10).PalID) - 1) _
    = Mid(PalletID_Now, 1, Len(PalletID_Now) - 1) _
    And RCT(RclIndex10).RouteCode = "R3M" _
    And RCT(RclIndex10).SendedRouteID <> "I" Then
        '-----この時、自R3Mが有り&自R3M未発報である-----
        '-----①発報ΦP-IDを次ΦP-ID エリアにセット-----
        CTR_NEXT_PALLET_ID_OLD = RCT(RclIndex10).NPID_R3M
        CTR_NEXT_PALLET_ID_NEW = ""
        '-----②経路制御TBLの自R3M発報記録の更新-----
        RCT(RclIndex10).SendedRouteID = "I"
        '-----経路制御TBLに自SFにR3Cが有るか?-----
        RclIndex11 = 0

```

【図 60】

```

Do Until RCT(RctIndex11).No < 1
  If Mid(RCT(RctIndex11).PalID, 1, Len(RCT(RctIndex11).PalID) - 1) _
    = Mid(PalletID_Now, 1, Len(PalletID_Now) - 1) _
  And RCT(RctIndex11).RouteCode = "R3C" Then
    '-----この時、自R3Cが有。パレット起動処理へ。-----'
    RctIndex1 = 0
    RctIndex2 = 0
    RctIndex3 = 0
    RctIndex4 = 0
    RctIndex5 = 0
    RctIndex6 = 0
    RctIndex7 = 0
    RctIndex8 = 0
    RctIndex9 = 0
    RctIndex10 = 0
    RctIndex11 = 0
    RctIndex12 = 0
    RctIndex13 = 0
    RctIndex14 = 0
    RctIndex15 = 0
    RctIndex16 = 0
    RctIndex17 = 0
    RctIndex18 = 0
    RctIndex19 = 0
    RctIndex20 = 0
    Exit Sub
  End If
  RctIndex11 = RctIndex11 + 1
Loop
'この時、経路制御TBLに自SFにR3Cが存在しないから
'-----経路制御TBLの自R3M発報記録のクリア-----'
RCT(RctIndex10).SendedRouteID = ""
'自SFのR3Mが作用する全てのR3Cの経路制御情報テーブルの発報記録をオン
'RctIndex10:自R3Mのレコードを指示するポインタ
'RctIndex18:自R3Mのレコードの、作用R3C情報を指示するポインタ
'RctIndex19:他R3Cのレコードを指示するポインタ
RctIndex18 = 0
Do Until RctIndex18 = 10
  If RCT(RctIndex10).SendedR3C_W03ID(RctIndex18) = "" Then Exit Do
  RctIndex19 = 0
  Do Until RCT(RctIndex19).No < 1
    If RCT(RctIndex19).PalID = RCT(RctIndex10).SendedR3C_W03ID(RctIndex18) _
      And RCT(RctIndex19).RouteCode = "R3C" _
      And RCT(RctIndex19).NPID_R3C = RCT(RctIndex10).SendedR3C_W04ID(RctIndex18) Then
      RCT(RctIndex19).SendedRouteID = "1"
    End If
    RctIndex19 = RctIndex19 + 1
  Loop
  RctIndex18 = RctIndex18 + 1
Loop
'-----パレット起動処理へ。-----'
RctIndex1 = 0
RctIndex2 = 0
RctIndex3 = 0

```

【図 6 1】

```
RctIndex4 = 0
RctIndex5 = 0
RctIndex6 = 0
RctIndex7 = 0
RctIndex8 = 0
RctIndex9 = 0
RctIndex10 = 0
RctIndex11 = 0
RctIndex12 = 0
RctIndex13 = 0
RctIndex14 = 0
RctIndex15 = 0
RctIndex16 = 0
RctIndex17 = 0
RctIndex18 = 0
RctIndex19 = 0
RctIndex20 = 0
Exit Sub
End If
RctIndex10 = RctIndex10 + 1
Loop
-----<(自SFにR3Cが有り&自SFの全R3C発報済?)or自R3Cが無し>-----
' RctIndex = 12 自R3C指示ポインタ
' RctIndex2 = 13 自R3C指示ポインタ
' RctIndex3 = 14 自SFの全R3C発報済CHK用カウンタ
Do Until RCT(RctIndex12).No < 1
  If RCT(RctIndex12).PalID = PalletID_Now _
    And RCT(RctIndex12).RouteCode = "R3C" Then
    GoTo BOX_ALL_R3C_SENDED_CHK
  End If
  RctIndex12 = RctIndex12 + 1
Loop
' この時、自SFにR3Cが無し
RctIndex1 = 0
RctIndex2 = 0
RctIndex3 = 0
RctIndex4 = 0
RctIndex5 = 0
RctIndex6 = 0
RctIndex7 = 0
RctIndex8 = 0
RctIndex9 = 0
RctIndex10 = 0
RctIndex11 = 0
RctIndex12 = 0
RctIndex13 = 0
RctIndex14 = 0
RctIndex15 = 0
RctIndex16 = 0
RctIndex17 = 0
RctIndex18 = 0
RctIndex19 = 0
```

【図 6 2】

```

RctIndex20 = 0
Call OnErrorPRC
Exit Sub
'この時、自R3Cがあり。自SFの全R3C発報済CHKへ
BOX_ALL_R3C_SENDED_CHK:
  Do Until RCT(RctIndex13).No < 1
    If RCT(RctIndex13).PalID = PalletID_Now _
      And RCT(RctIndex13).RouteCode = "R3C" _
      And RCT(RctIndex13).SendedRouteID = "I" Then
      RctIndex14 = RctIndex14 + 1
      If RCT(RctIndex13).NextPalleIs = RctIndex14 Then
        'この時、自SFの全R3Cが発報済である
        RctIndex1 = 0
        RctIndex2 = 0
        RctIndex3 = 0
        RctIndex4 = 0
        RctIndex5 = 0
        RctIndex6 = 0
        RctIndex7 = 0
        RctIndex8 = 0
        RctIndex9 = 0
        RctIndex10 = 0
        RctIndex11 = 0
        RctIndex12 = 0
        RctIndex13 = 0
        RctIndex14 = 0
        RctIndex15 = 0
        RctIndex16 = 0
        RctIndex17 = 0
        RctIndex18 = 0
        RctIndex19 = 0
        RctIndex20 = 0
        Call OnErrorPRC
        Exit Sub
      End If
    End If
    RctIndex13 = RctIndex13 + 1
  Loop
'経路制御情報テーブルの自R3D&自R3M発報記録のクリア
  Do Until RCT(RctIndex15).No < 1
    If RCT(RctIndex15).PalID = PalletID_Now _
      And (RCT(RctIndex15).RouteCode = "R3D" _
      Or RCT(RctIndex15).RouteCode = "R3M") Then
      RCT(RctIndex15).SendedRouteID = ""
    End If
    RctIndex15 = RctIndex15 + 1
  Loop
  Do Until RCT(RctIndex16).No < 1
    If RCT(RctIndex16).PalID = PalletID_Now _
      And RCT(RctIndex16).RouteCode = "R3C" _
      And RCT(RctIndex16).SendedRouteID <> "I" Then
      '-----①発報ΦP-IDを次ΦP-ID エリアにセット-----'
      CTR_NEXT_PALLET_ID_OLD = RCT(RctIndex16).NPID_R3C
      CTR_NEXT_PALLET_ID_NEW = ""
    End If
    RctIndex16 = RctIndex16 + 1
  Loop

```

【図 63】

```

'-----②発報記録をオン
      RCT (RctIndex16). SendedRouteID = '1'
'-----この時、パレット起動処理へ。-----
      RctIndex1 = 0
      RctIndex2 = 0
      RctIndex3 = 0
      RctIndex4 = 0
      RctIndex5 = 0
      RctIndex6 = 0
      RctIndex7 = 0
      RctIndex8 = 0
      RctIndex9 = 0
      RctIndex10 = 0
      RctIndex11 = 0
      RctIndex12 = 0
      RctIndex13 = 0
      RctIndex14 = 0
      RctIndex15 = 0
      RctIndex16 = 0
      RctIndex17 = 0
      RctIndex18 = 0
      RctIndex19 = 0
      RctIndex20 = 0
      Exit Sub

    End If
    RctIndex16 = RctIndex16 + 1
  Loop
  Call OnErrorPRC
  Exit Sub
End Sub

'*****初期起動パレットIDセット処理*****
Sub SetFirstPalletID ()
  RctIndex = 0
  Do Until RCT (RctIndex). No < 0
    If RCT (RctIndex). Row = 1 And RCT (RctIndex). Col = 1 Then
      CTR_NEXT_PALLET_ID_NEW = Left (RCT (RctIndex). PalID, Len (RCT (RctIndex). PalID) - 1) & '4'
      RctIndex = 0
      Exit Sub
    End If
    RctIndex = RctIndex + 1
  Loop
End Sub

'*****パレット起動処理*****
Sub RunPallets ()
  Rem CALLPLT
  If CTR_NEXT_PALLET_ID_OLD = '@%PALID@' Then %n      Call @%PALID@ %n      Exit Sub %n      End If
End Sub

'*****エラー終了処理*****
Sub OnErrorPRC ()
  END
End Sub

'*****経路情報テーブル作成処理*****
Sub MakeRuleControlTable ()
  RctIndex = 0

```

【図 6 4】

```
Open "syubetu_out.txt" For Input As #1
Do Until EOF(1)
    Line Input #1, InputRCT
    splitEx InputRCT, vbTab, ForSplit(0)
    RCT(RctIndex).No = ForSplit(0)
    RCT(RctIndex).SFID = ForSplit(1)
    RCT(RctIndex).Row = ForSplit(2)
    RCT(RctIndex).Col = ForSplit(3)
    RCT(RctIndex).PalID = ForSplit(4)
    RCT(RctIndex).SubNo = ForSplit(5)
    RCT(RctIndex).RouteCode = ForSplit(6)
    RCT(RctIndex).NextPallets = ForSplit(7)
    RCT(RctIndex).NPID_R3R = ForSplit(8)
    RCT(RctIndex).NPID_R3M = ForSplit(9)
    RCT(RctIndex).NPID_R3D = ForSplit(10)
    RCT(RctIndex).NPID_R3C = ForSplit(11)
    RCT(RctIndex).SendedRouteID = ForSplit(12)
    RctIndex1 = 0
    Do Until RctIndex1 = 10
        splitEx ForSplit(13 + RctIndex1), ",", ForSplit2(0)
        RCT(RctIndex).SendedR3C_W03ID(RctIndex1) = ForSplit2(0)
        RCT(RctIndex).SendedR3C_W04ID(RctIndex1) = ForSplit2(1)
        ForSplit2(0) = ""
        ForSplit2(1) = ""
        RctIndex1 = RctIndex1 + 1
    Loop
    RctIndex = RctIndex + 1
Loop
Close #1
InputRCT = ""
RctIndex = 0
RctIndex1 = 0
ForSplit(0) = ""
ForSplit(1) = ""
ForSplit(2) = ""
ForSplit(3) = ""
ForSplit(4) = ""
ForSplit(5) = ""
ForSplit(6) = ""
ForSplit(7) = ""
ForSplit(8) = ""
ForSplit(9) = ""
ForSplit(10) = ""
ForSplit(11) = ""
ForSplit(12) = ""
ForSplit(13) = ""
ForSplit(14) = ""
ForSplit(15) = ""
ForSplit(16) = ""
ForSplit(17) = ""
ForSplit(18) = ""
ForSplit(19) = ""
ForSplit(20) = ""
ForSplit(21) = ""
```



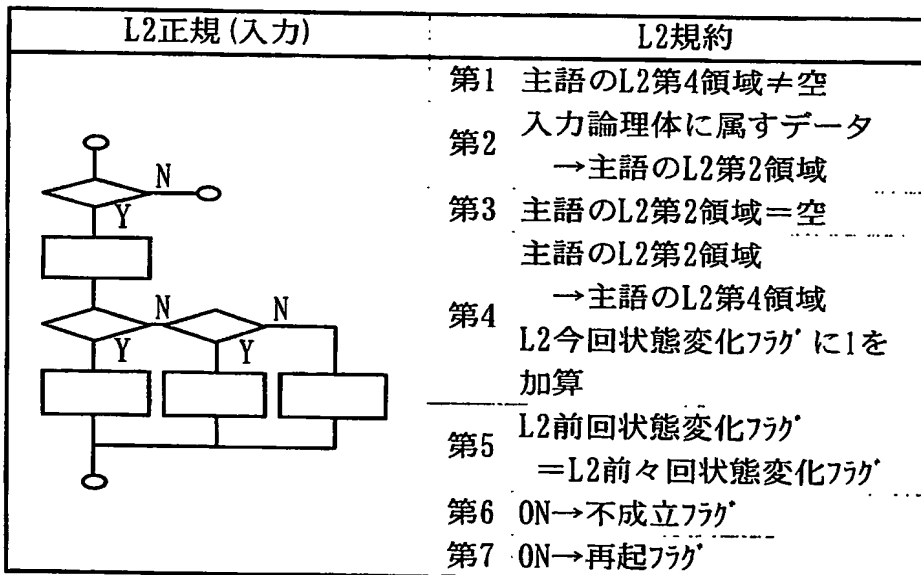
【図 6 5】

```

ForSplit(22) = ""
End Sub
'*****SPLIT関数*****
Public Sub splitEx(sText As String, sDelimiter As String, ByRef sResult() As String)
    Dim lLenText As Long
    Dim lPos As Long
    Dim lFindPos As Long
    Dim i As Long
    ReDim sResult(0)
    lLenText = Len(sText)
    If lLenText = 0 Then
        Exit Sub
    End If
    i = 0
    For lPos = 1 To lLenText
        lFindPos = InStr(lPos, sText, sDelimiter)
        If lFindPos = 0 Then lFindPos = lLenText + 1
        ReDim Preserve sResult(i)
        sResult(i) = Mid$(sText, lPos, lFindPos - lPos)
        i = i + 1
        lPos = lFindPos
    Next
    If Mid$(sText, lLenText, 1) = sDelimiter Then
        ReDim Preserve sResult(i)
        sResult(i) = ""
    End If
End Sub

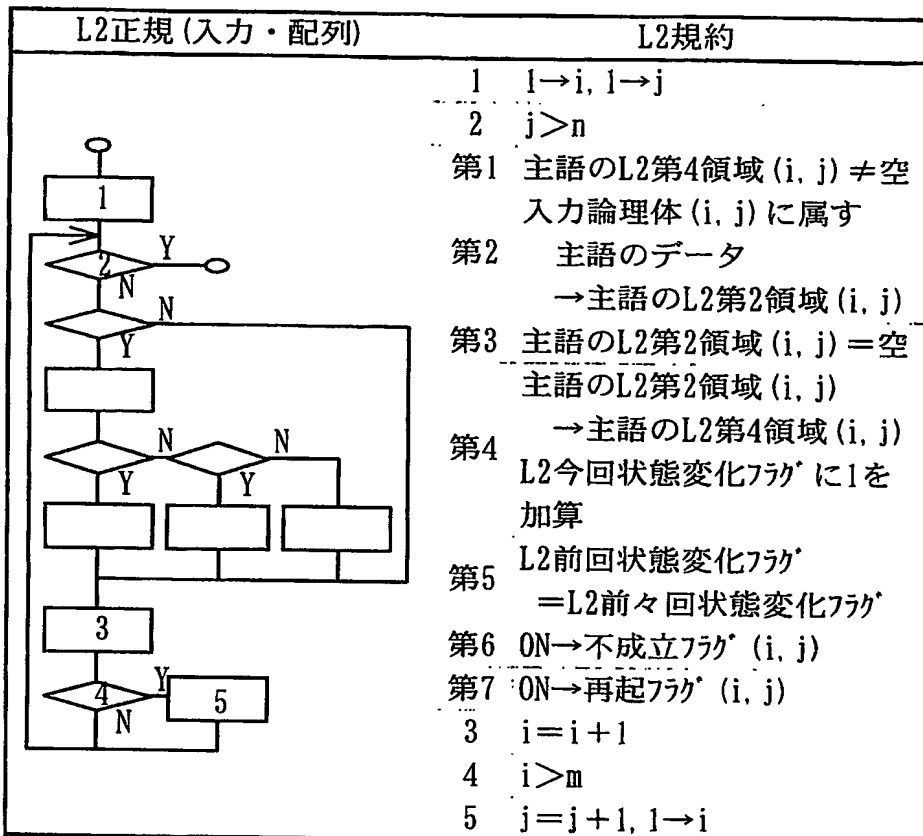
```

【図 6 6】



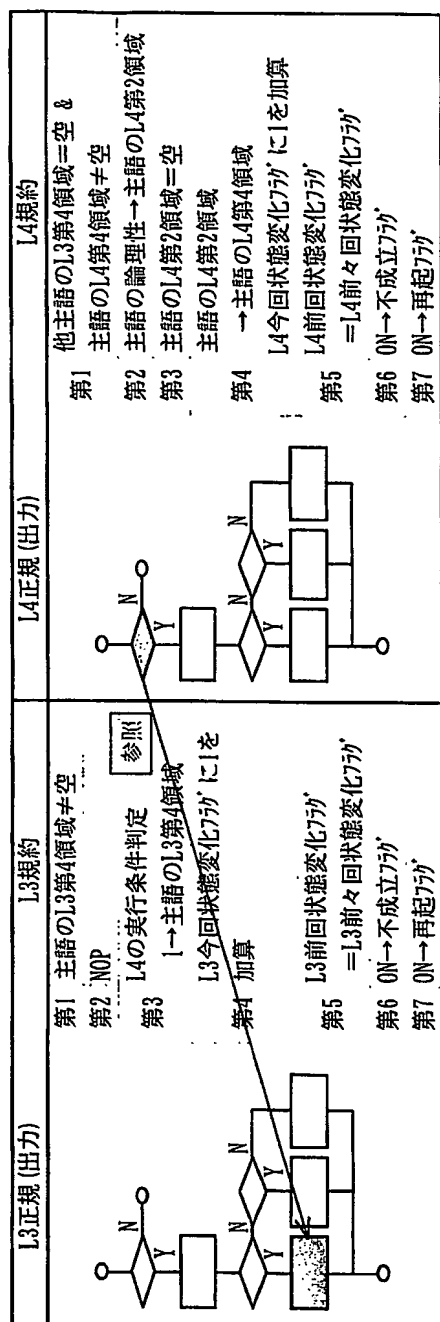
定義規則図 1 : 主語が正規でその属性が入力のベクトル

【圖 6 7】



定義規則図 2：主語が正規でその属性が入力，配列のベクトル

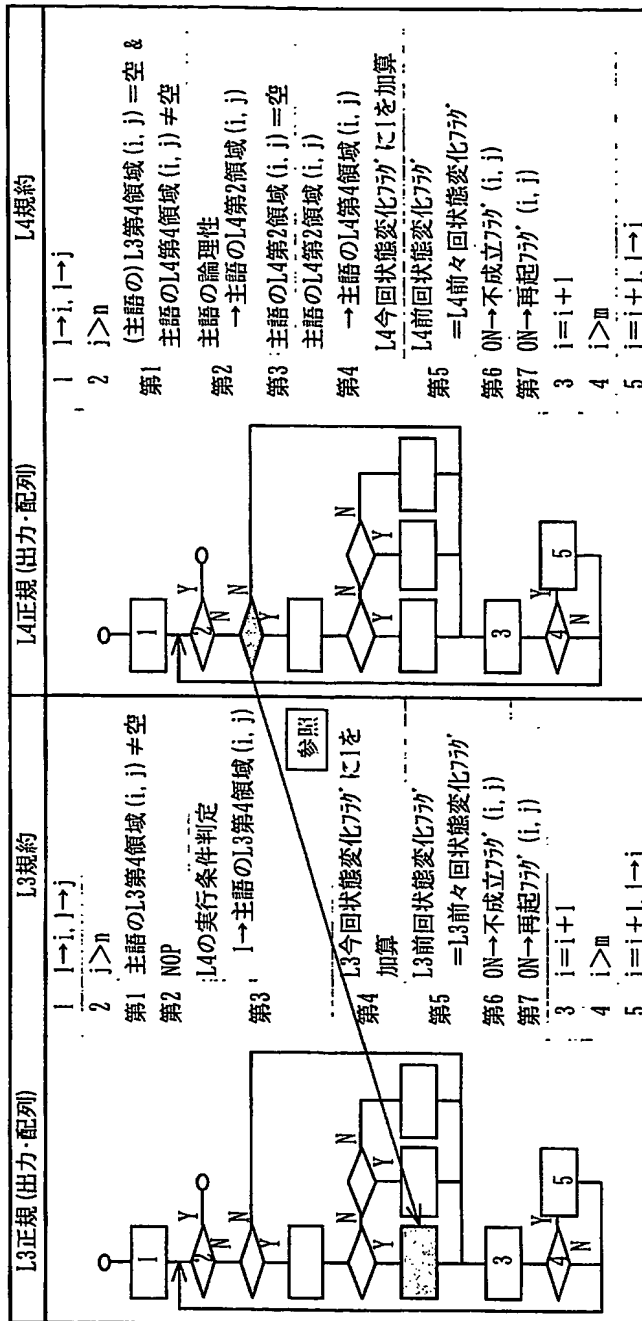
【図 68】



**定義規則図3：主語が正規でその属性が出力のベクトル**



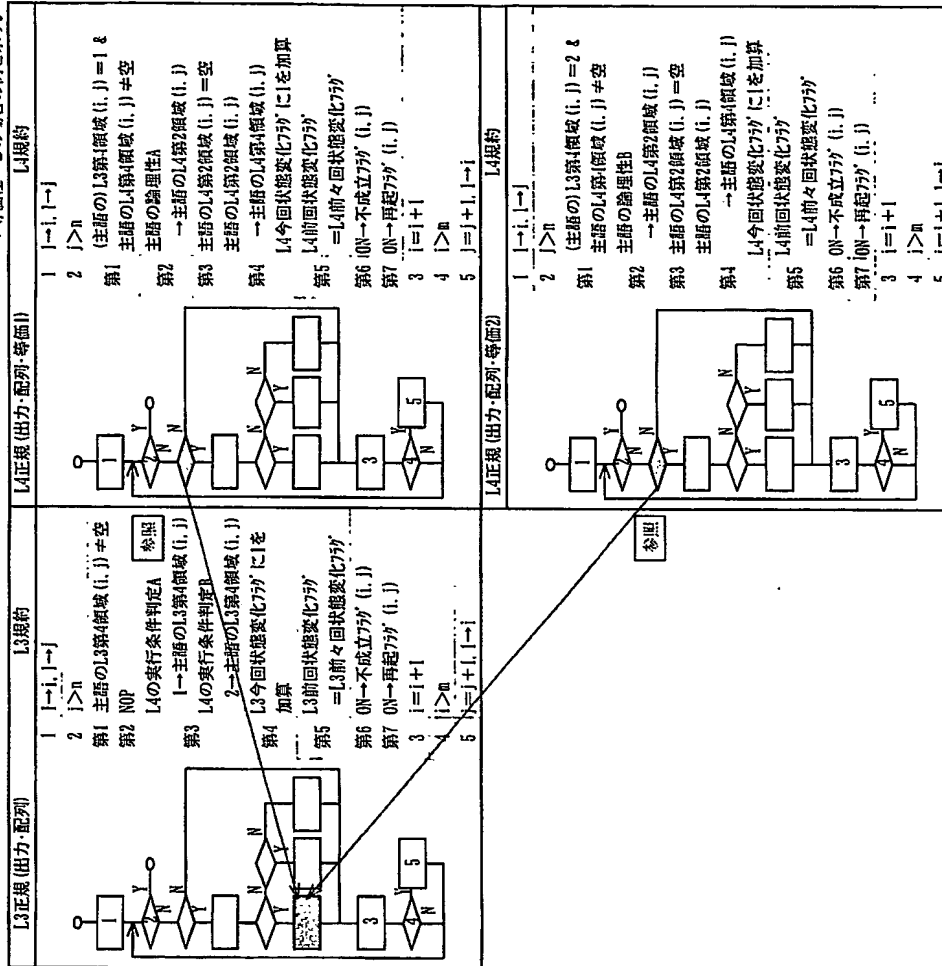
【図 70】



定義規則図 5: 主語が正規でその属性が出力、配列のベクトル

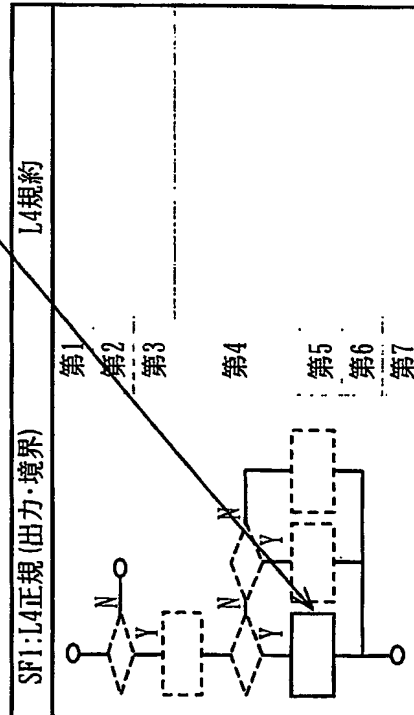
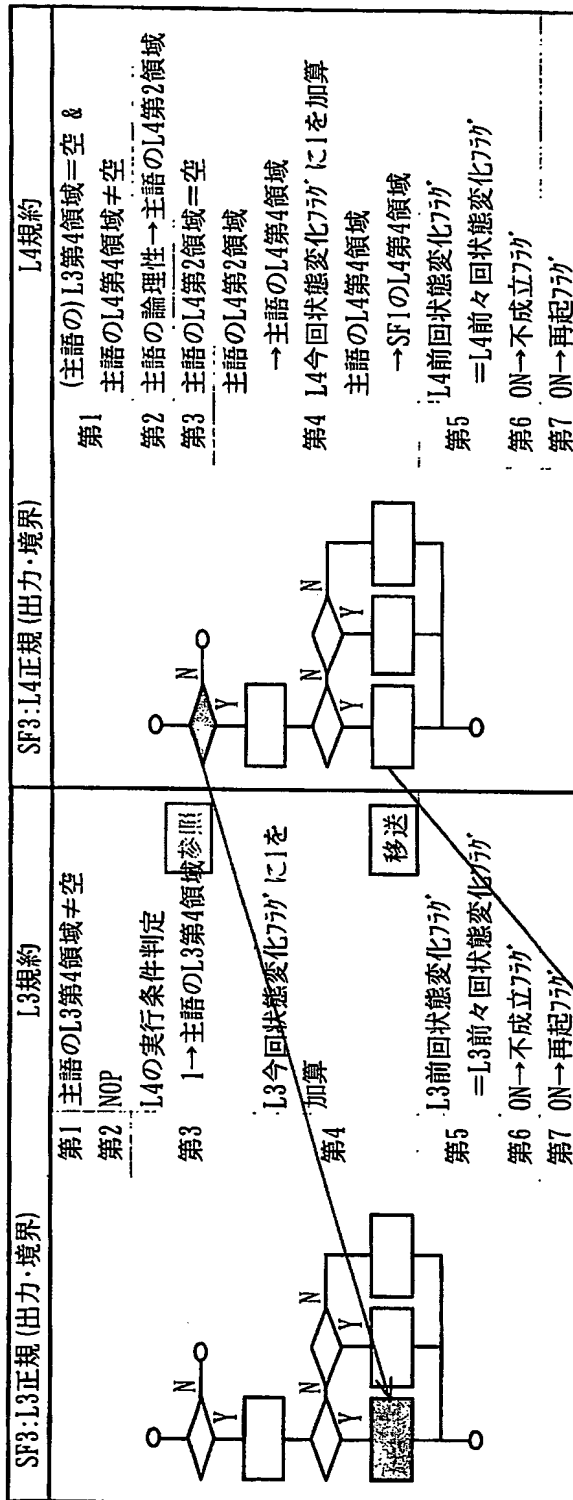
【図 7 1】

\*等価性=2の場合の例を示す。



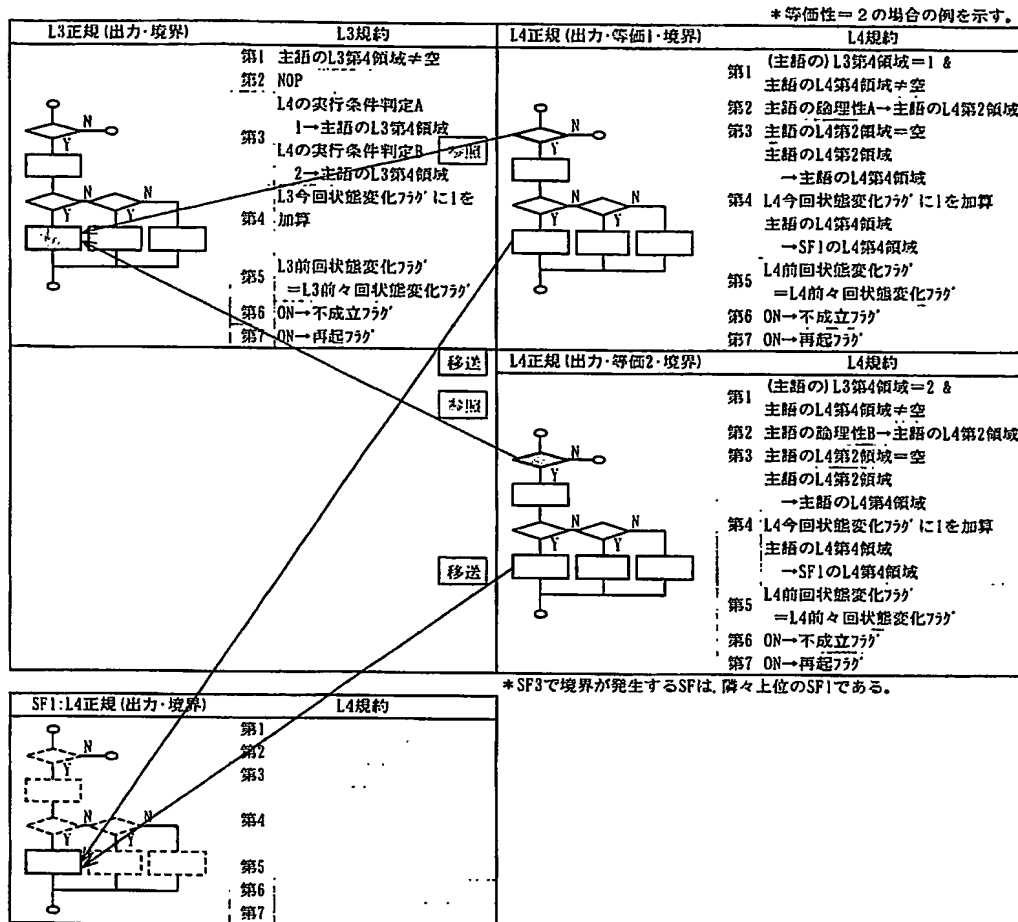
定義規則 6: 主語が正規でその属性が出力、等価、配列のベクトル

【図 72】



定義規則図 7: 主語が正規でその属性が出力、境界のベクトル

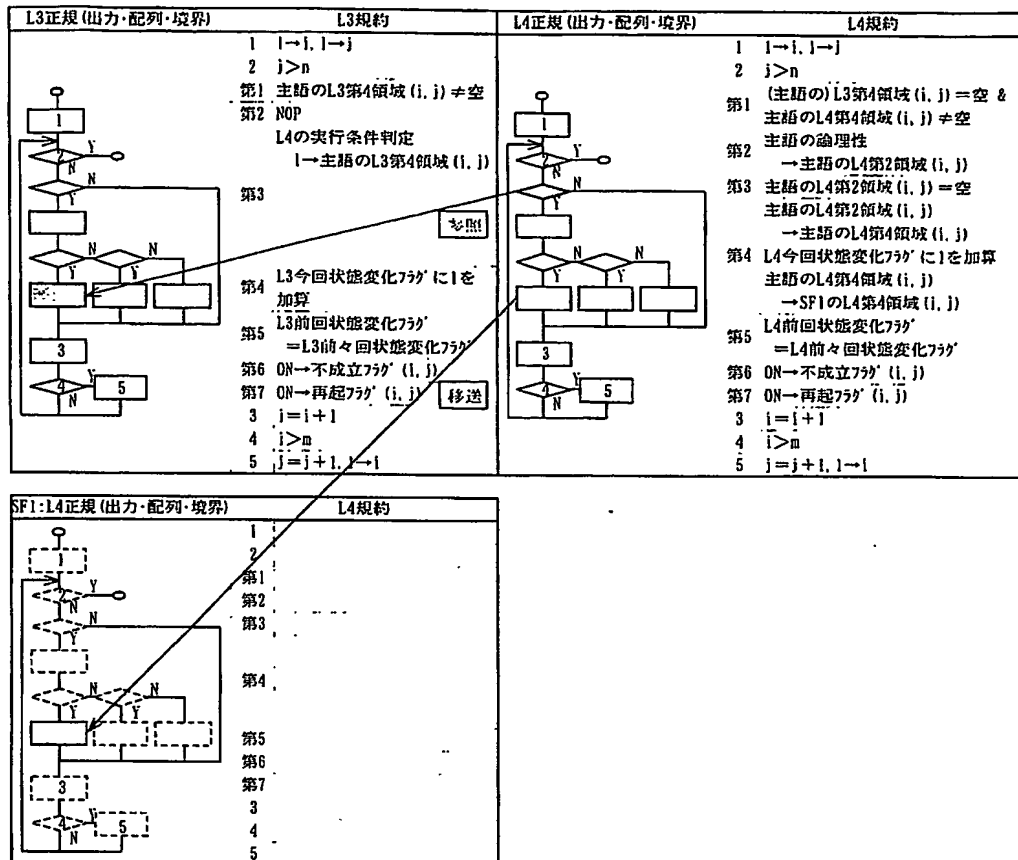
【図 73】



定義規則図 8 : 主語が正規でその属性が出力、等価、境界のベクトル

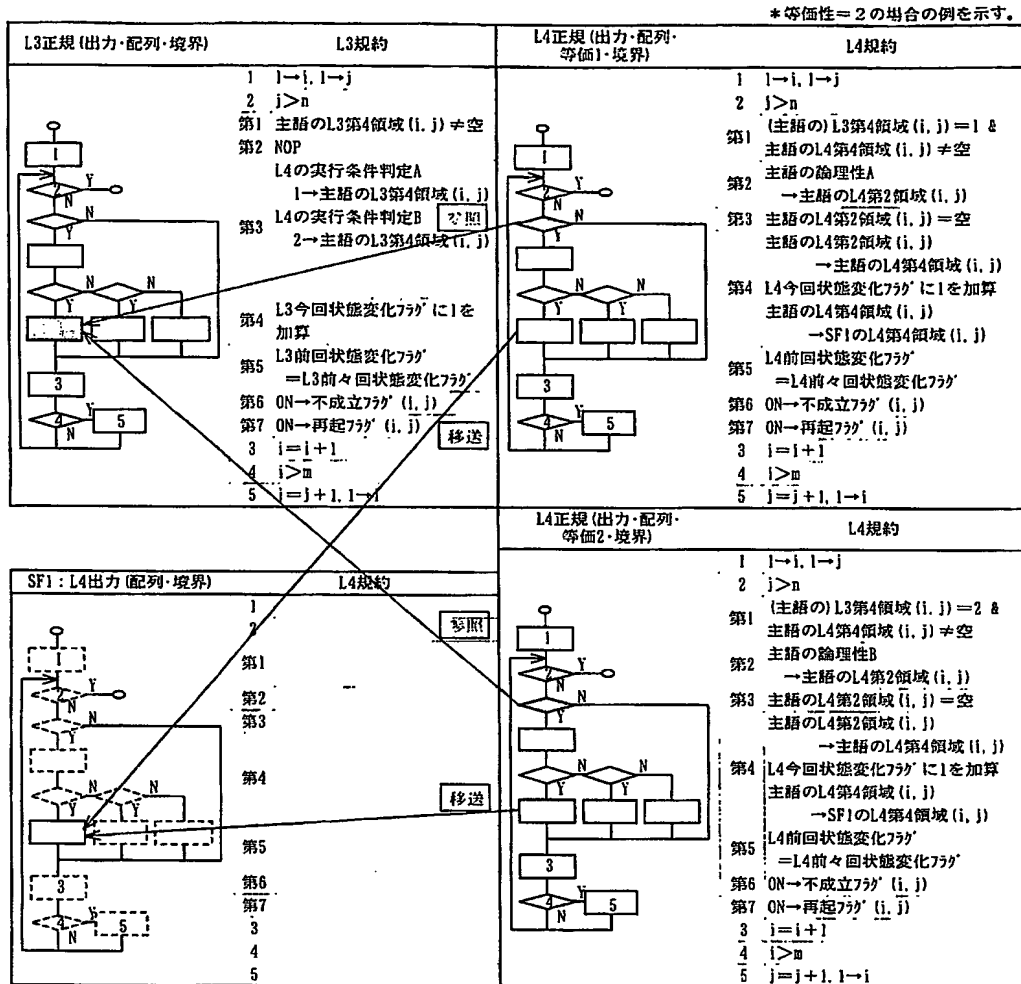


【図 7 4】



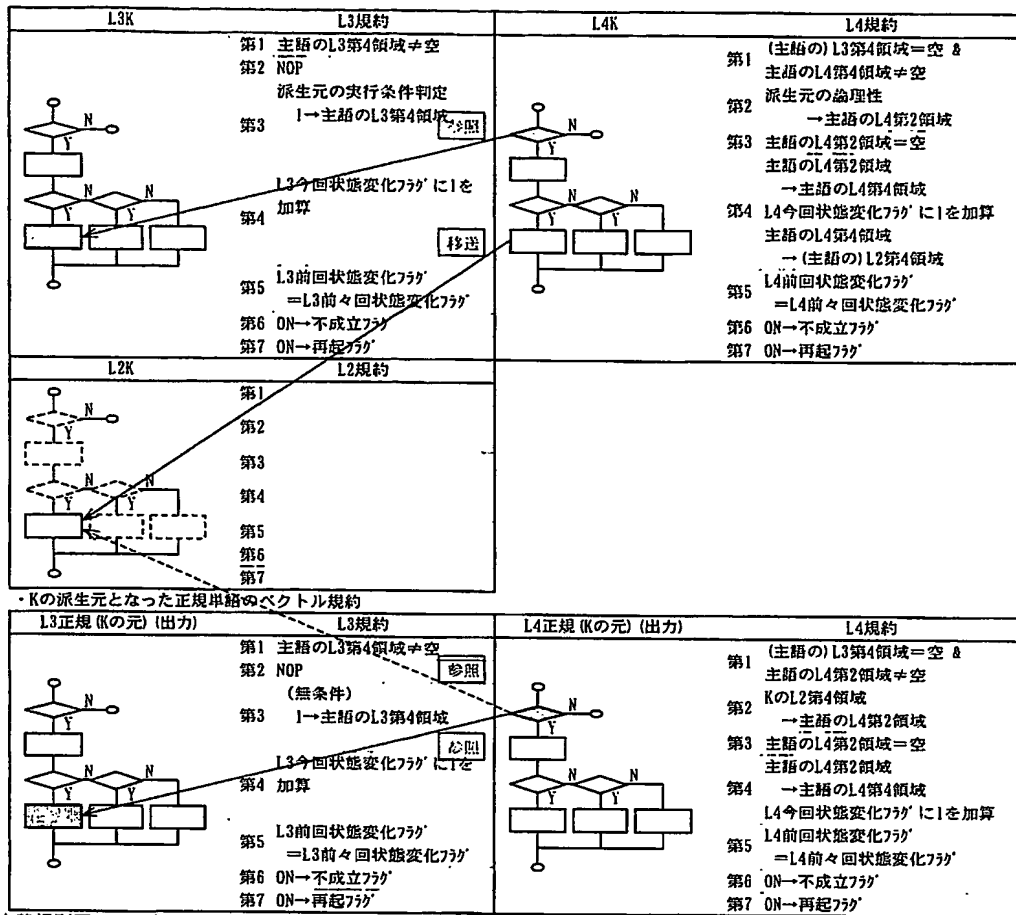
定義規則図 9 : 主語が正規でその属性が出力, 配列, 境界のベクトル

【図 75】



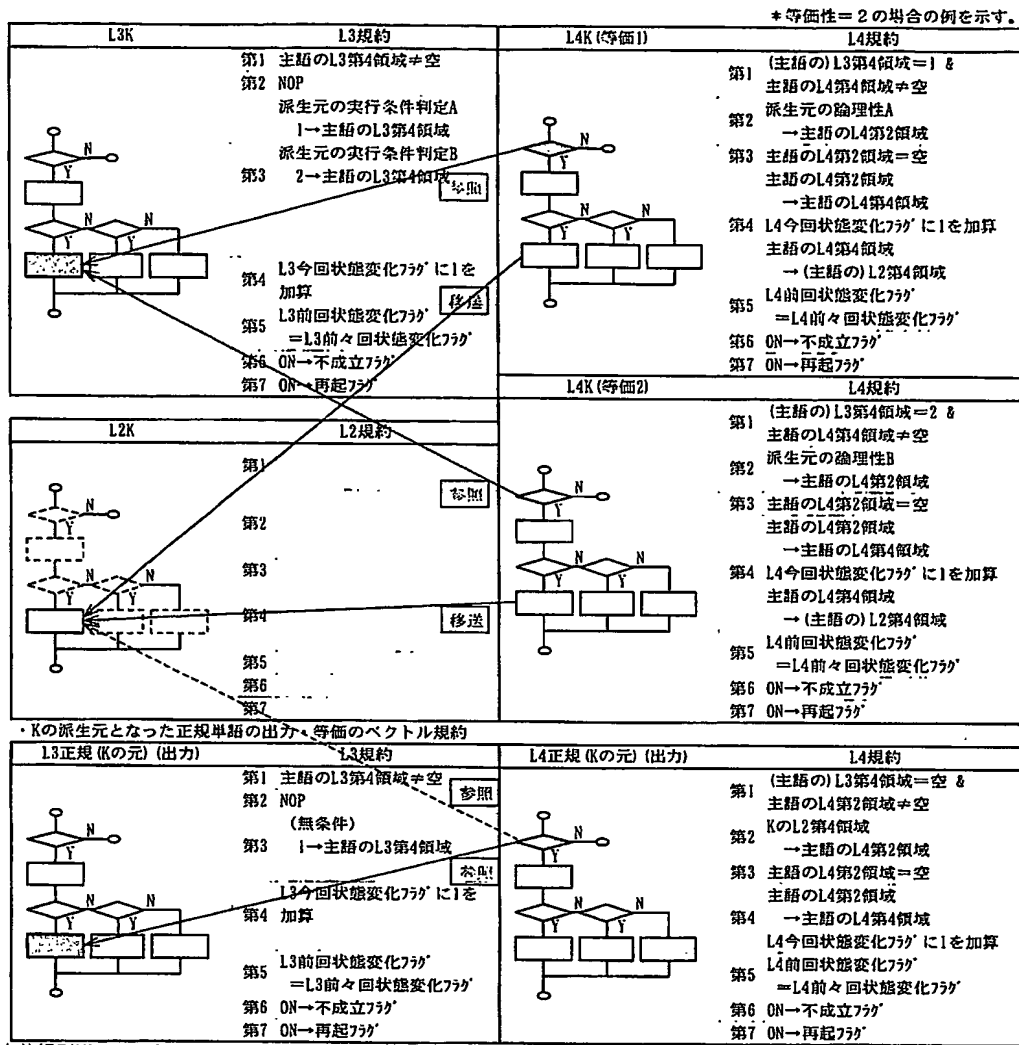
定義規則図 10 : 主語が正規でその属性が出力, 等価, 配列, 境界のベクトル

【図 76】

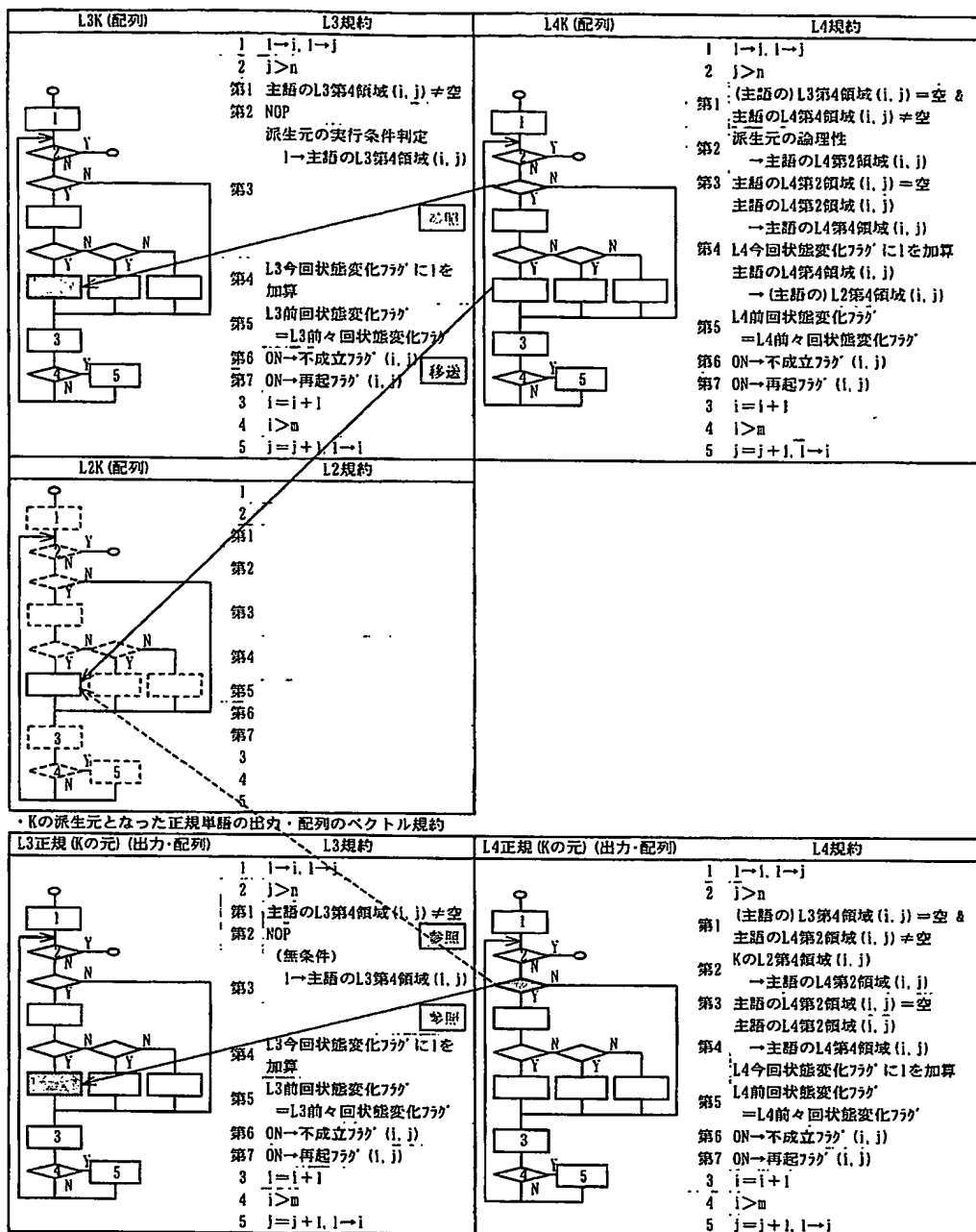


定義規則図 11 : 主語がKのベクトル

【図 77】

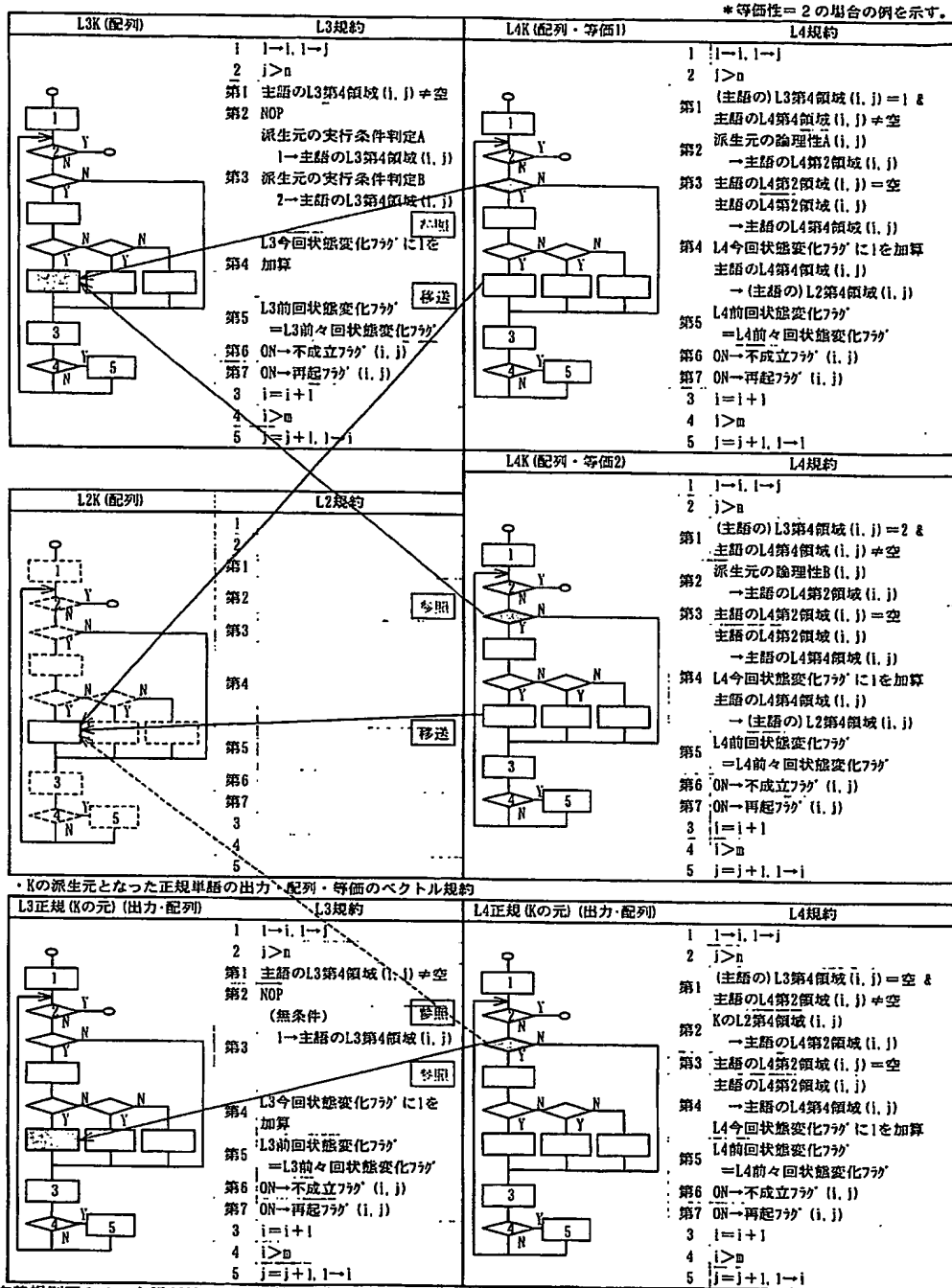


【图 78】



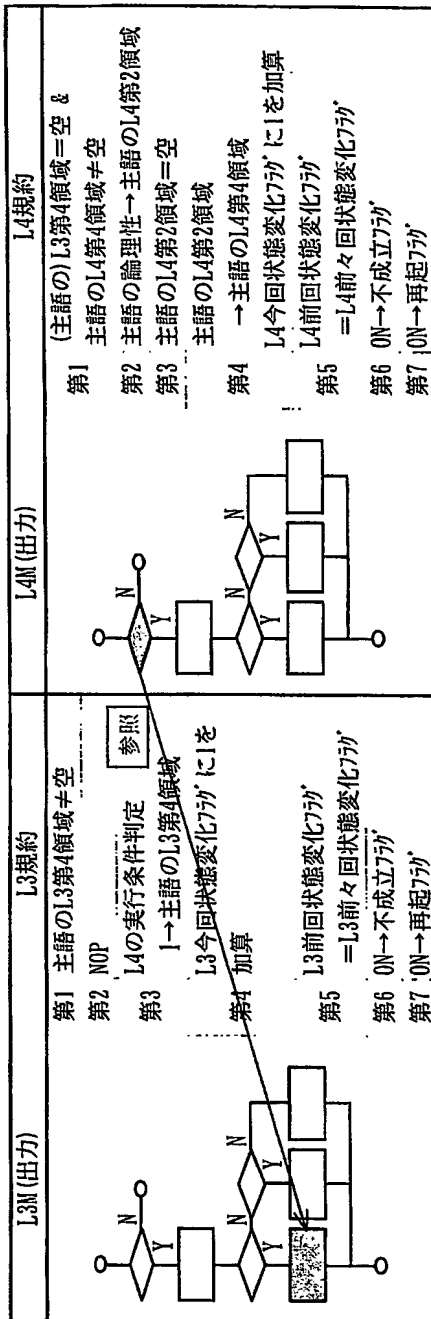
定義規則図 13 : 主語がKでその属性が配列のベクトル

【図 79】



定義規則図 14: 主語がKでその属性が等価, 配列のベクトル

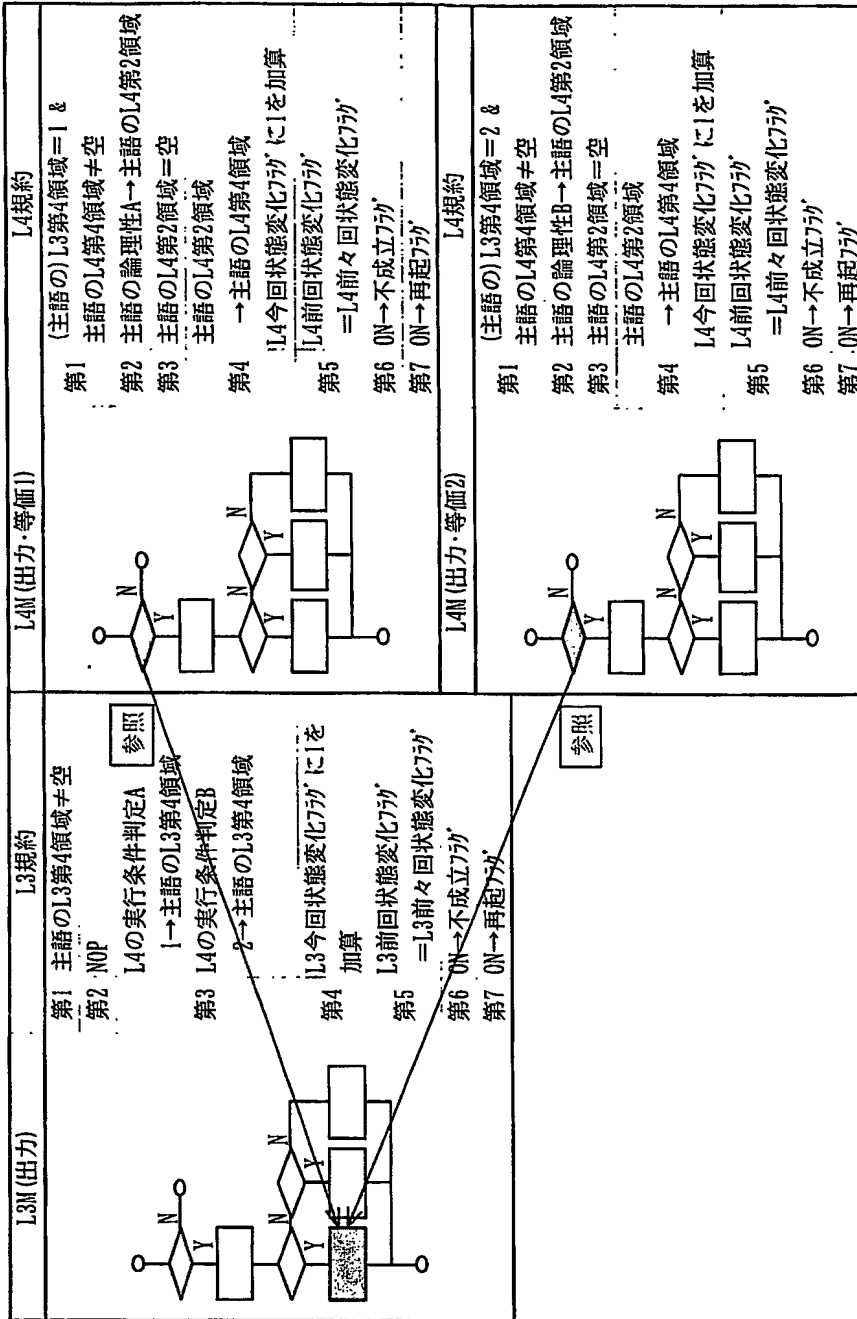
【圖 80】



定義規則図 15：主語がMでその属性が出力のベクトル

【図 81】

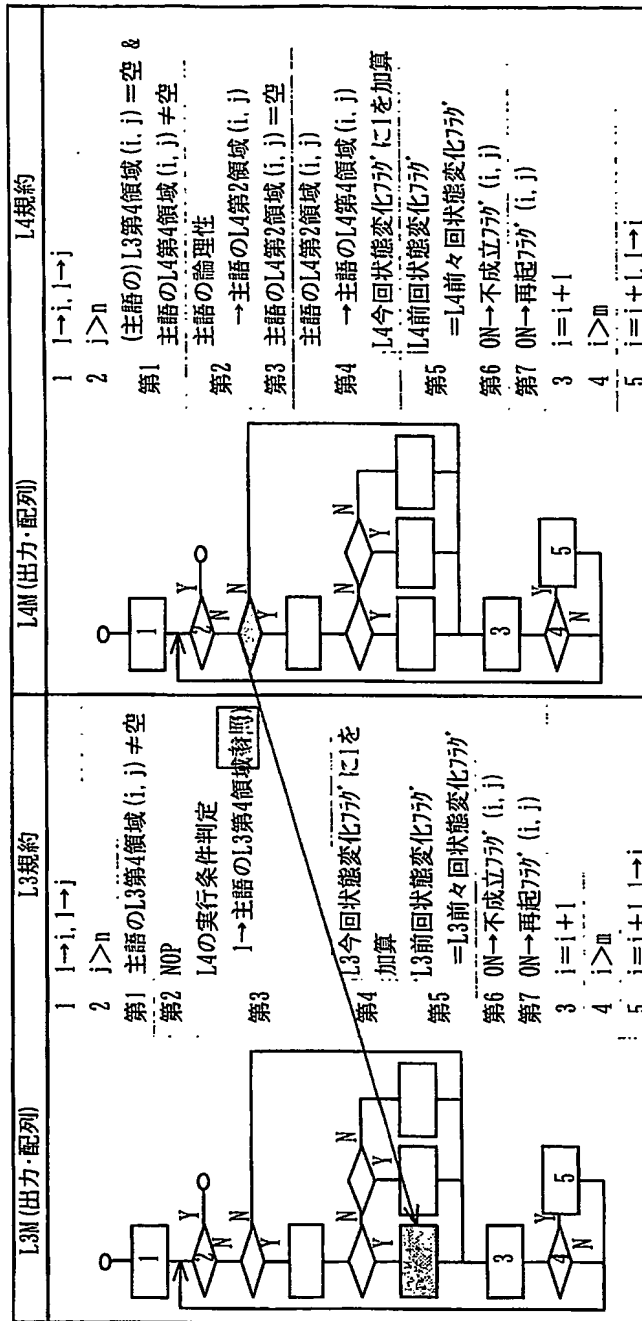
\* 等価性=2 の場合の例を示す。



定義規則図 16 : 主語がMでその属性が出力、等価のベクトル

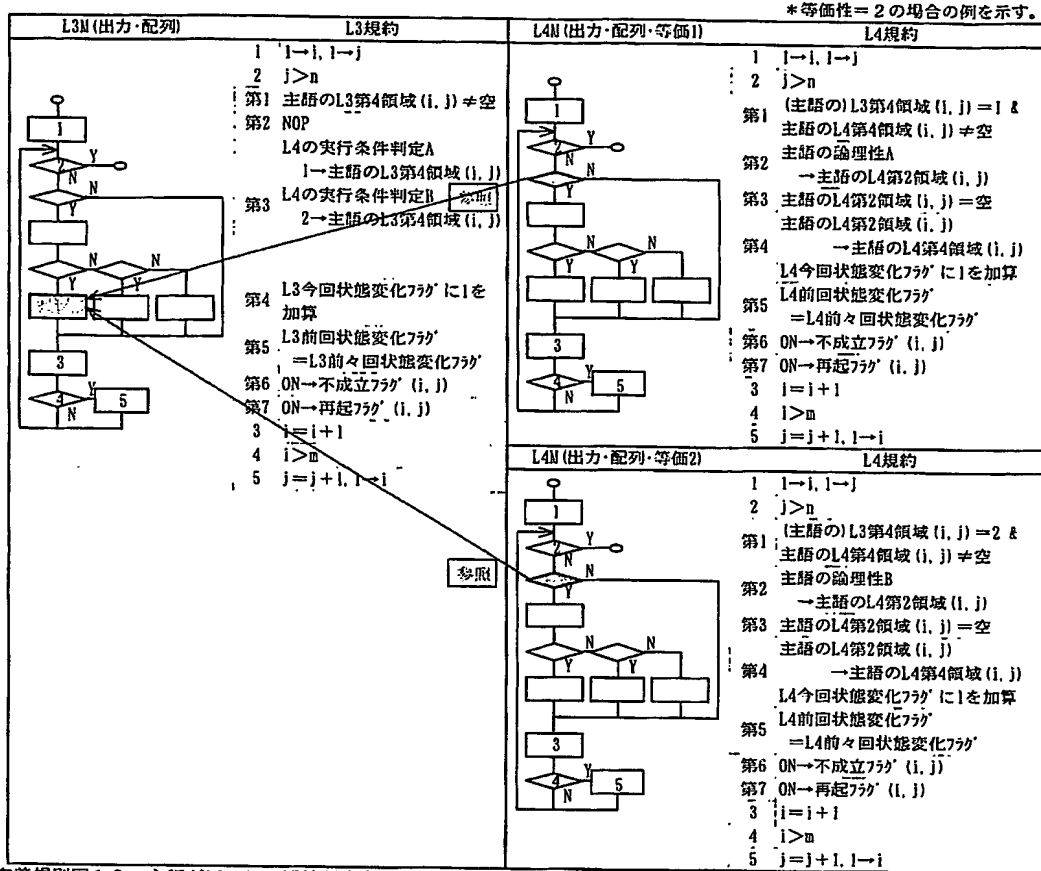


【図 8 2】



定義規則図 17: 主語がMでその属性が出力、配列のベクトル

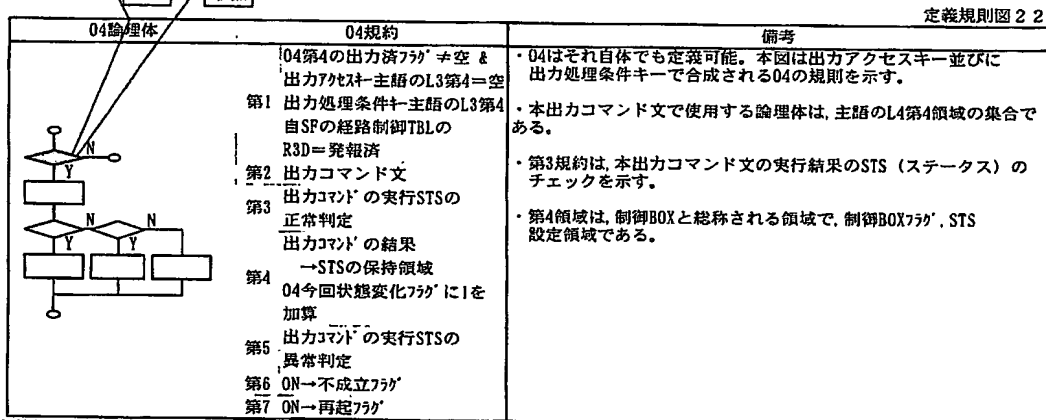
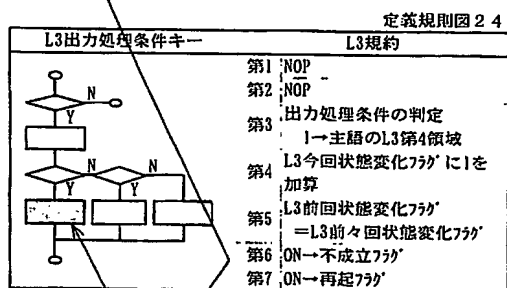
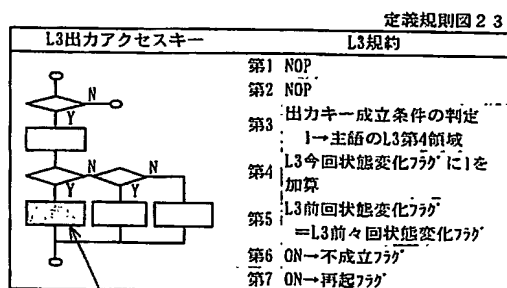
【図 83】



定義規則図 18 : 主語がMでその属性が出力、等価、配列のベクトル



【図 85】

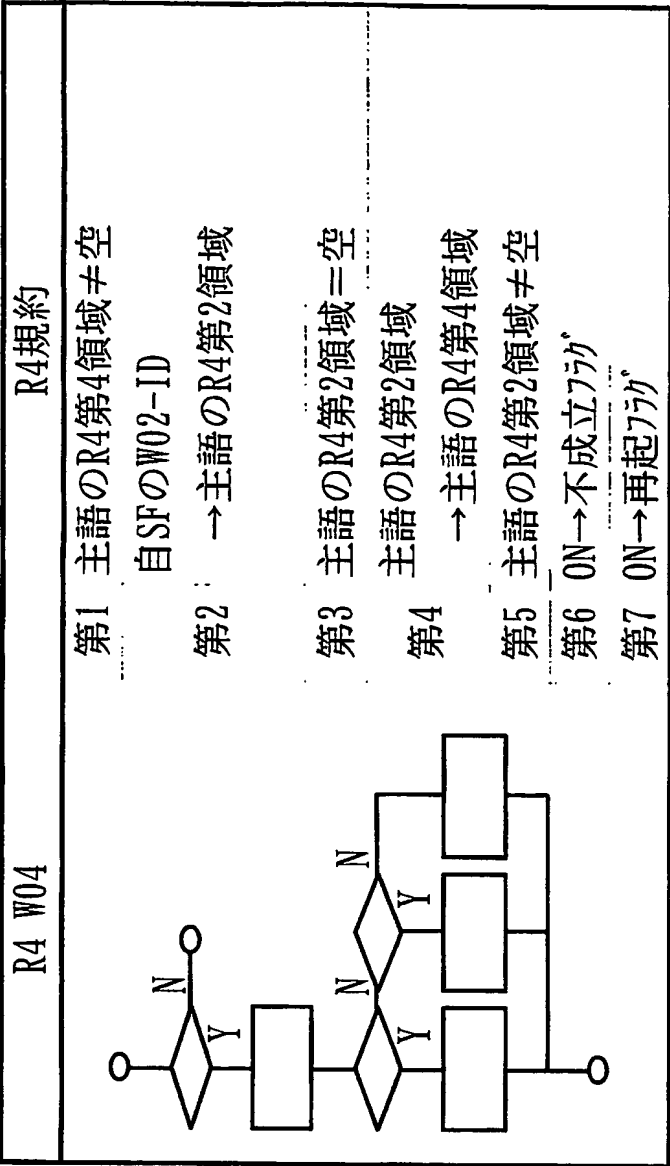


定義規則図 22 : 主語が論理体でその属性が出力のベクトル

定義規則図 23 : 主語がアクセスキーでその属性が出力のベクトル

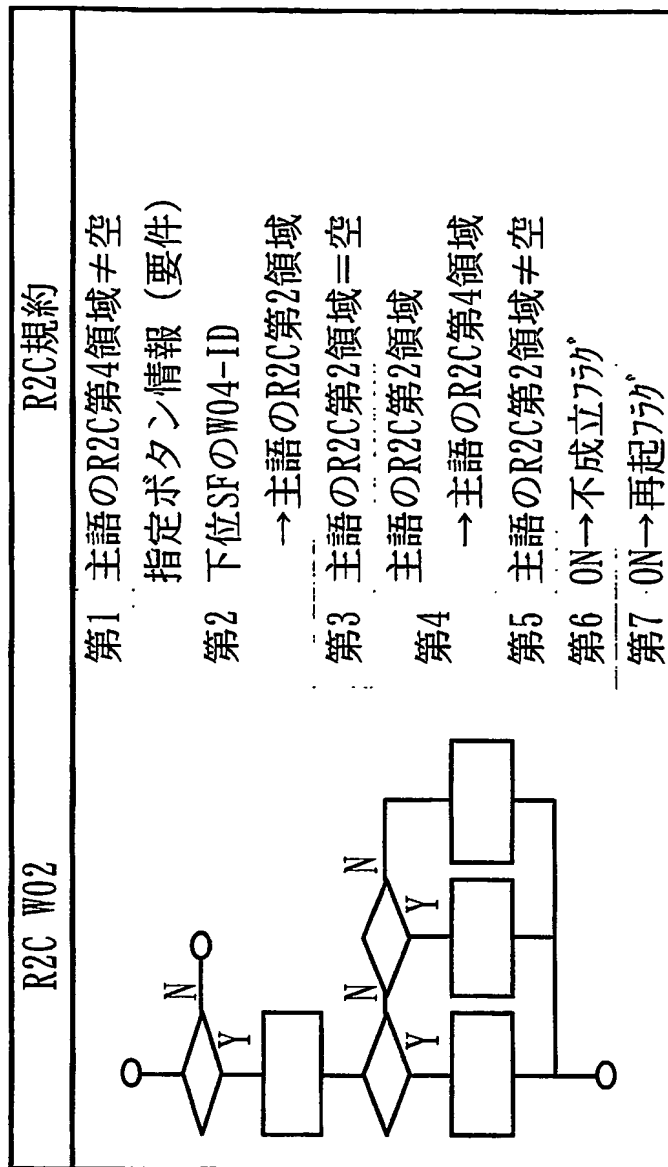
定義規則図 24 : 主語が処理条件キーでその属性が出力のベクトル

【図 86】



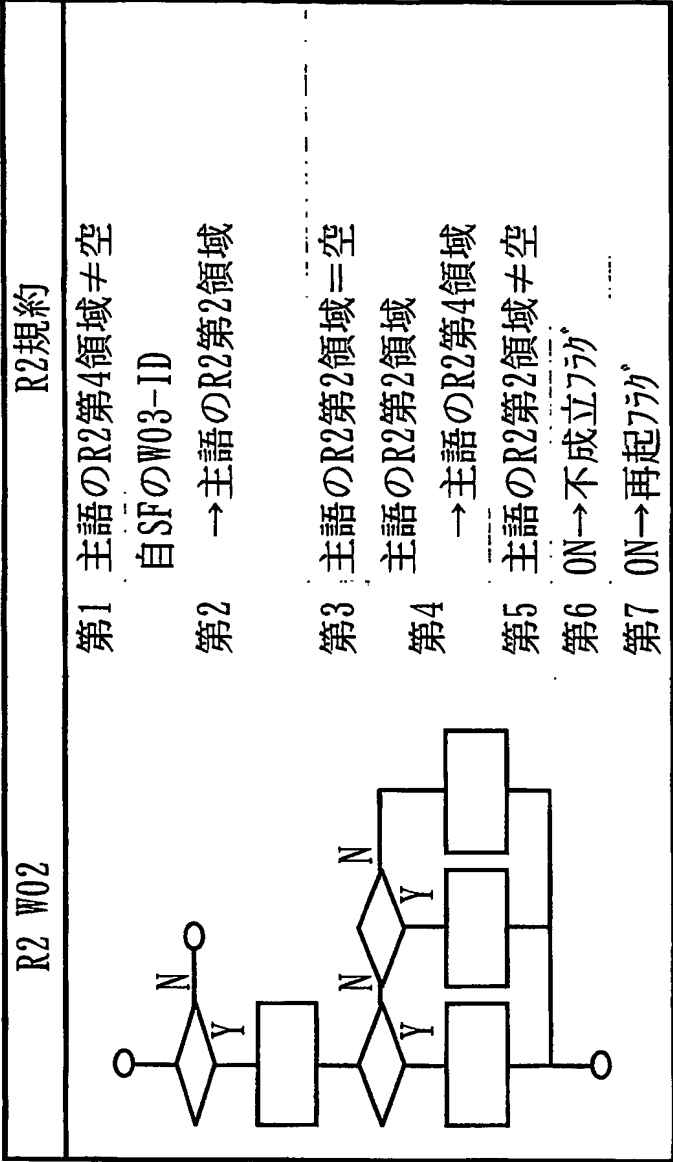
定義規則図 25: 主語がパレットW04でR4のベクトル

【図 8 7】



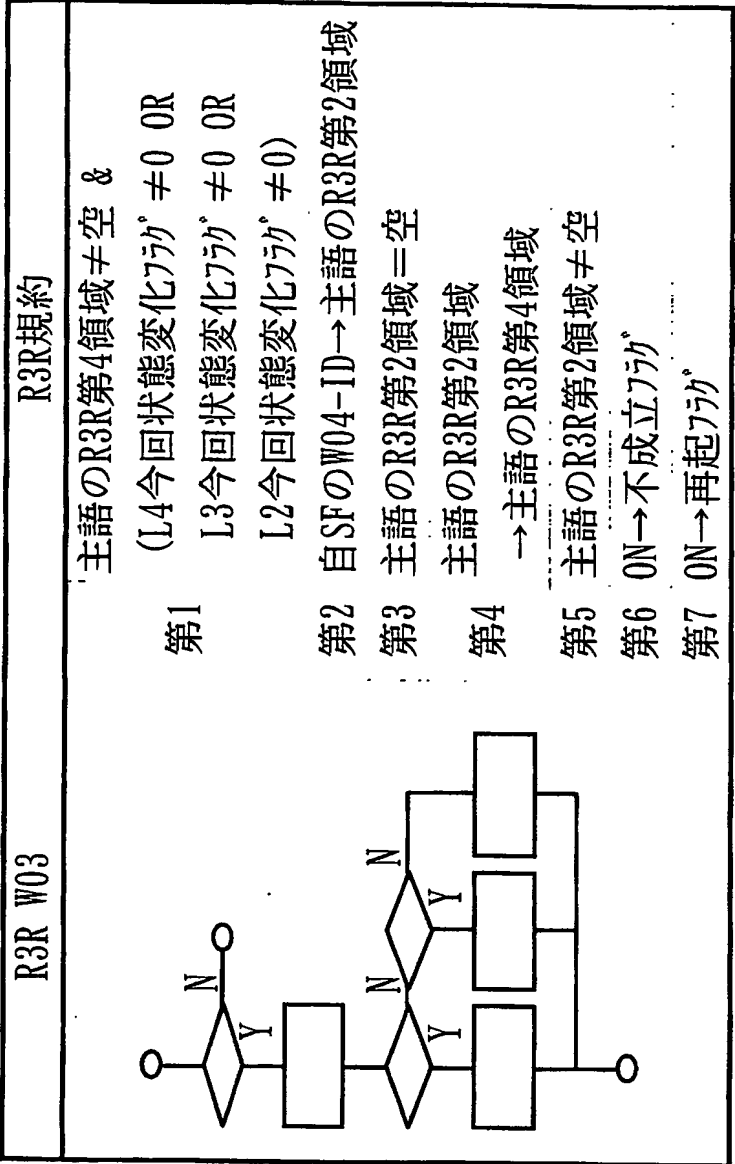
定義規則図26：主語がバレットW02でR2Cのベクトル

【図 88】



定義規則図 27：主語がパレットW02でR2のベクトル

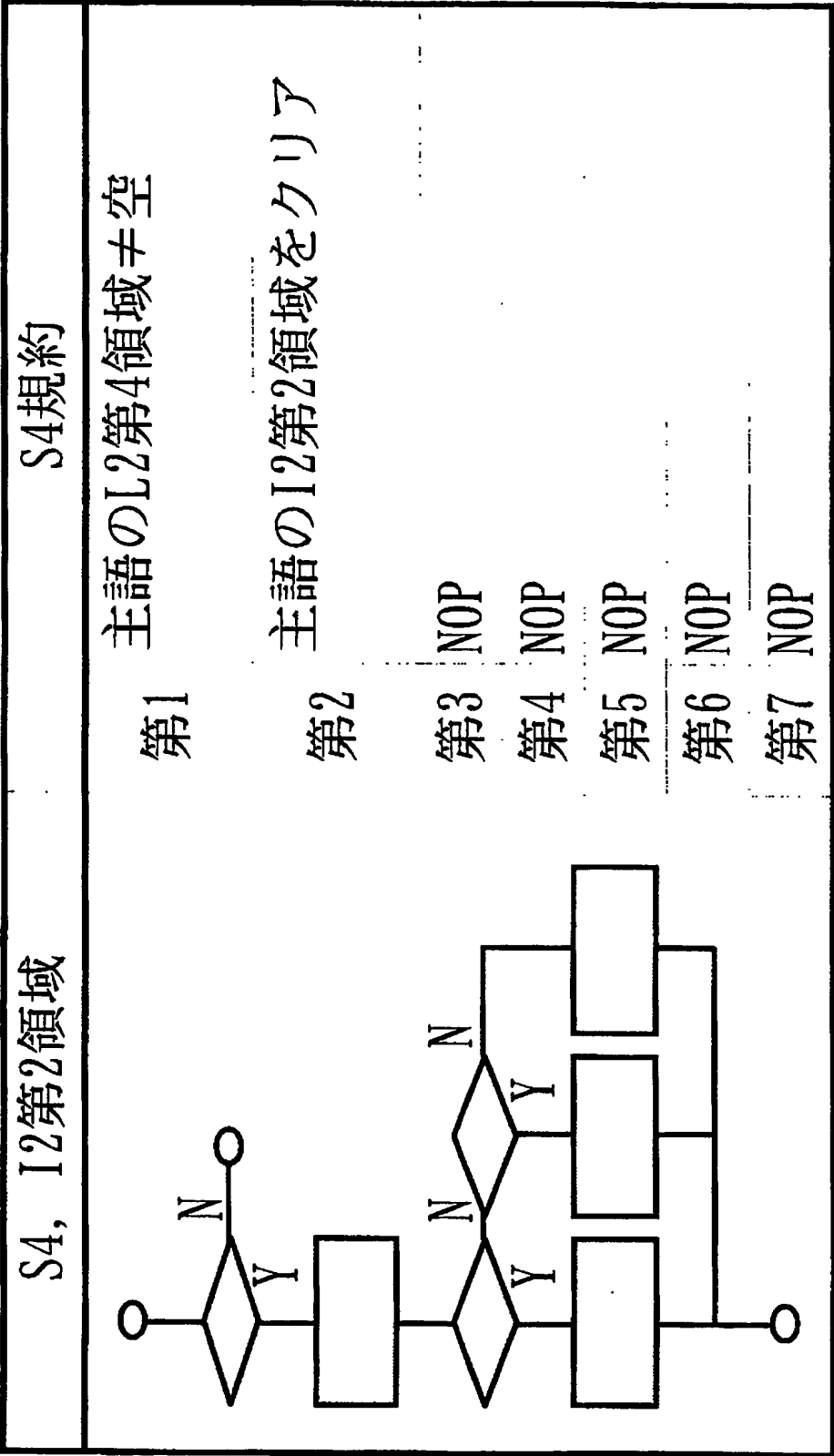
【図 89】



定義規則図 28：主語がパレットW03でR3Rのベクトル

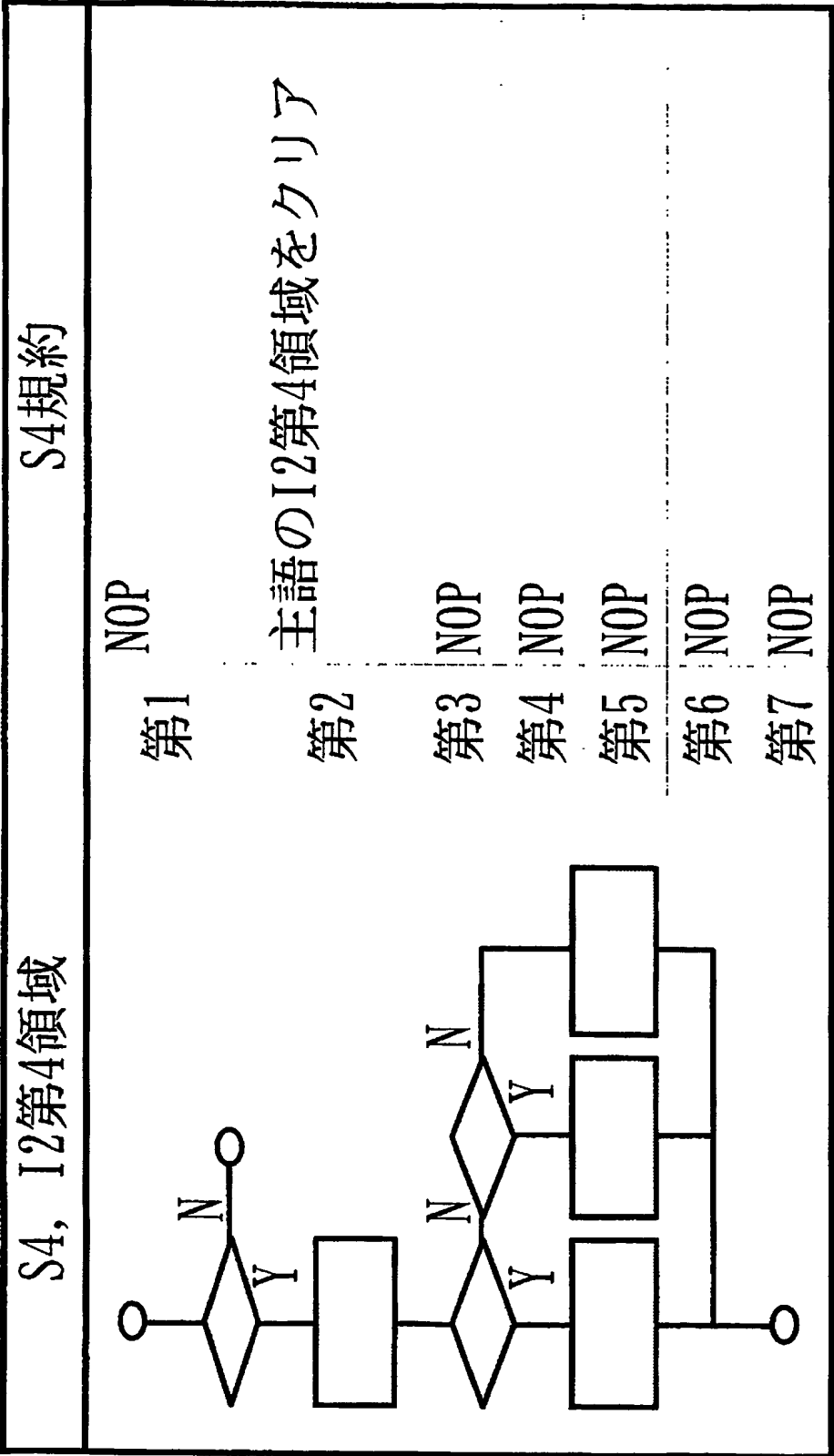


【図 90】



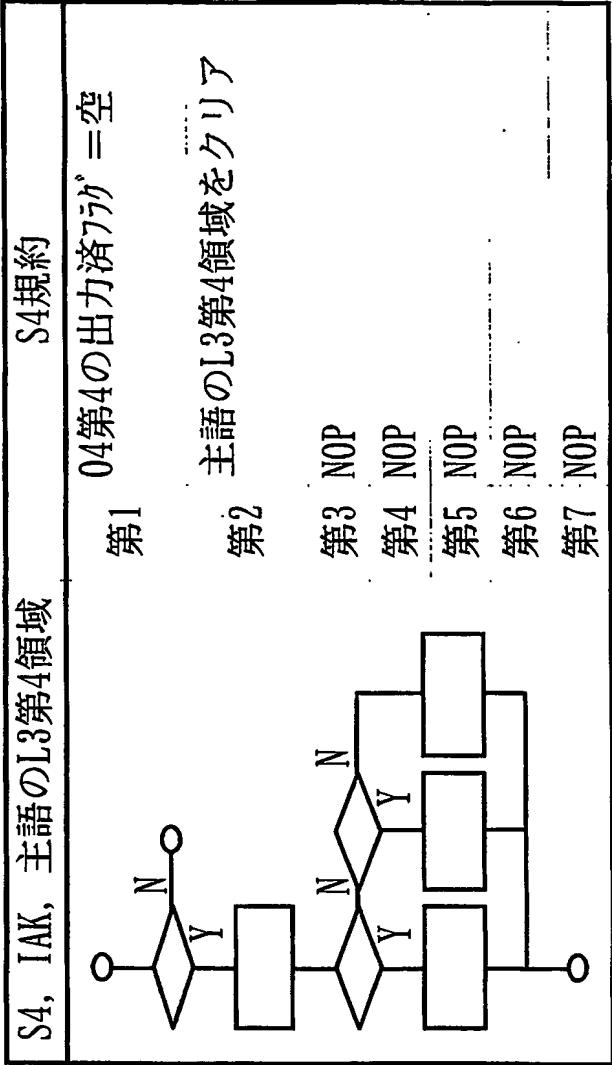
定義規則図 29：主語が I 2 第 2 領域の S 4 のベクトル

【図 9 1】



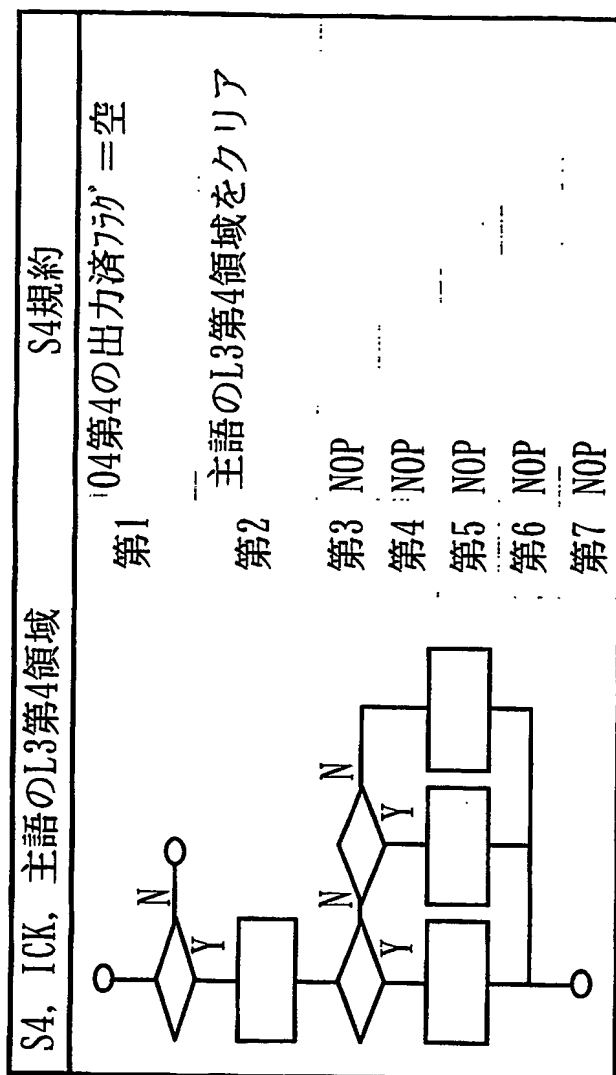
定義規則図 30 : 主語が I 2 第 4 領域の S 4 のベクトル

【図 9 2】



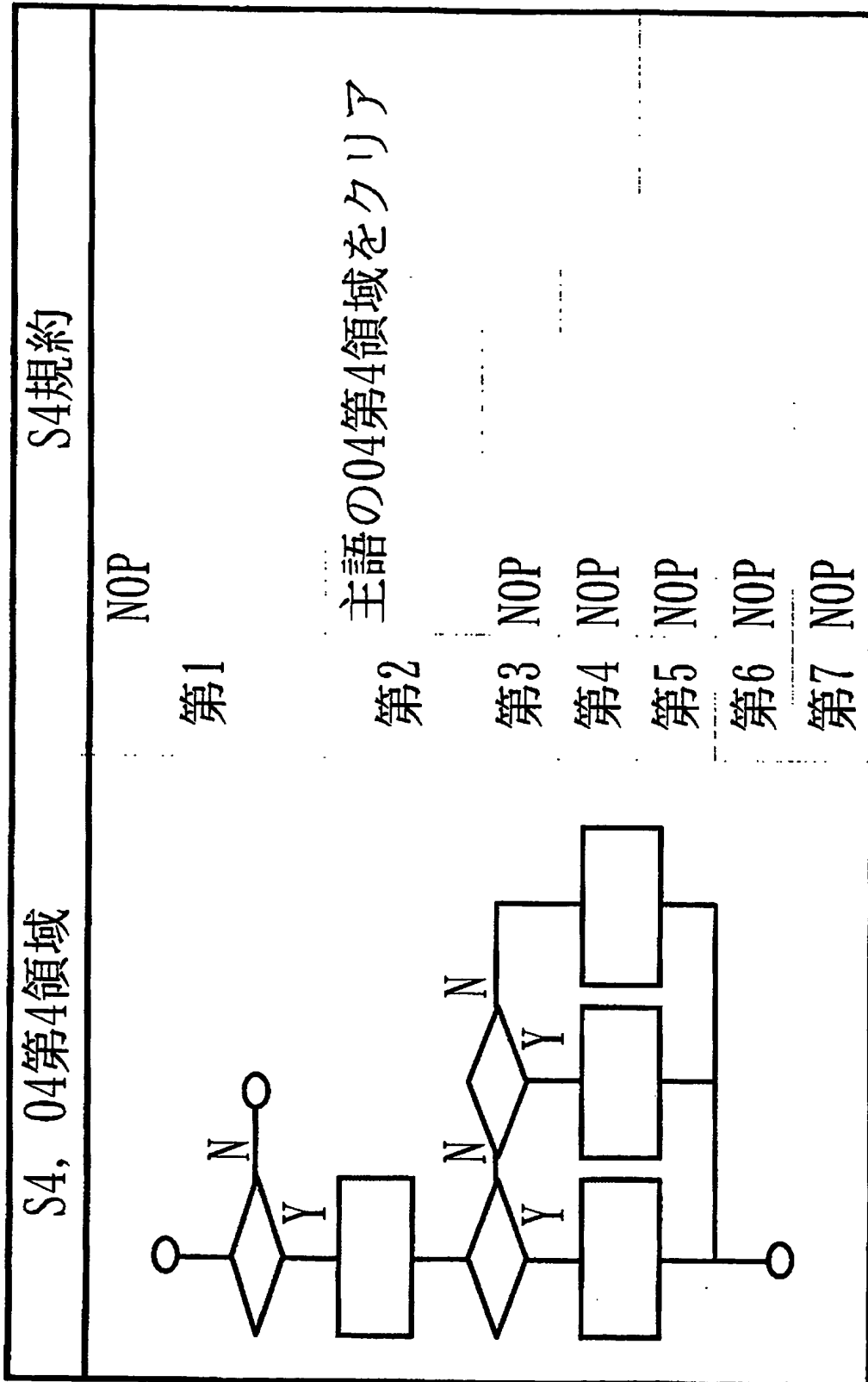
定義規則図 3 1：主語が入カアクセスキーのL 3 第4領域のS 4 のベクトル

【圖 9 3】



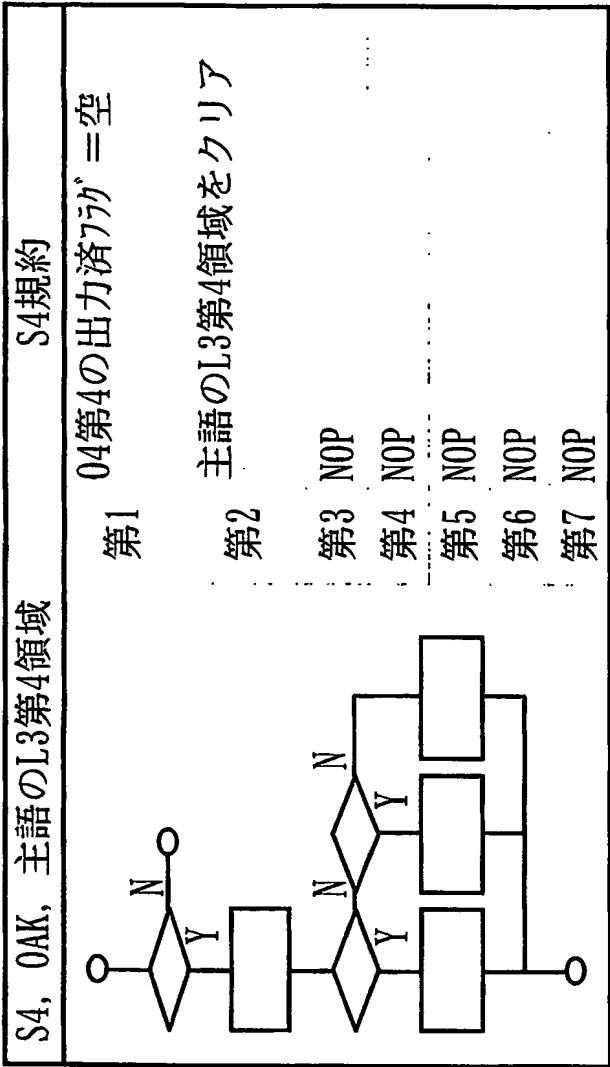
定義規則図 3.2 : 主語が入力処理条件キーの L3 第 4 領域の S4 のベクトル

【図 94】



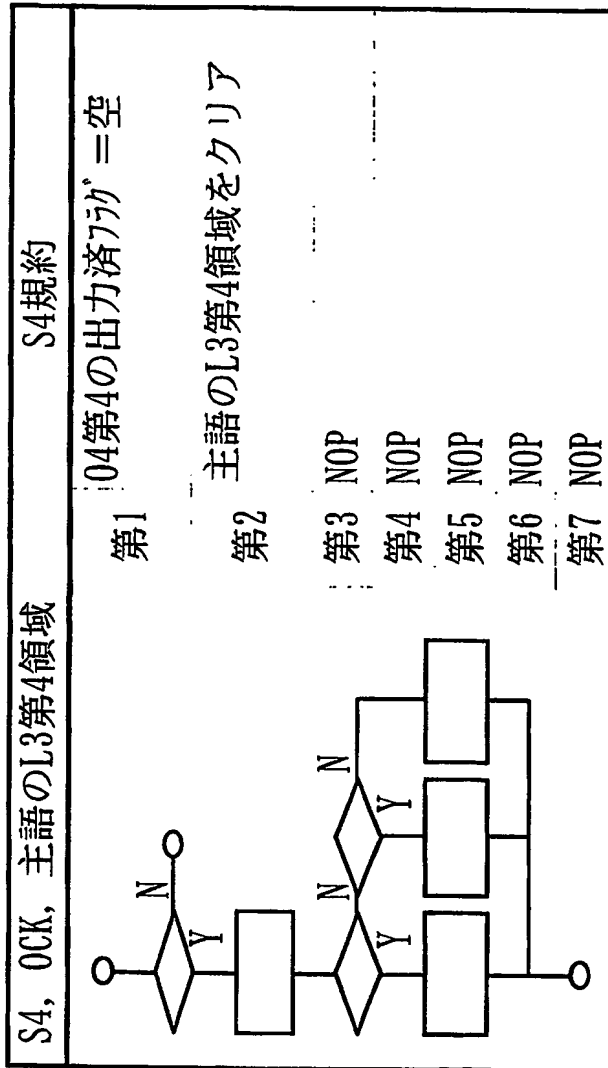
定義規則図 33 : 主語が 04 第4領域の S4 のベクトル

【図 95】



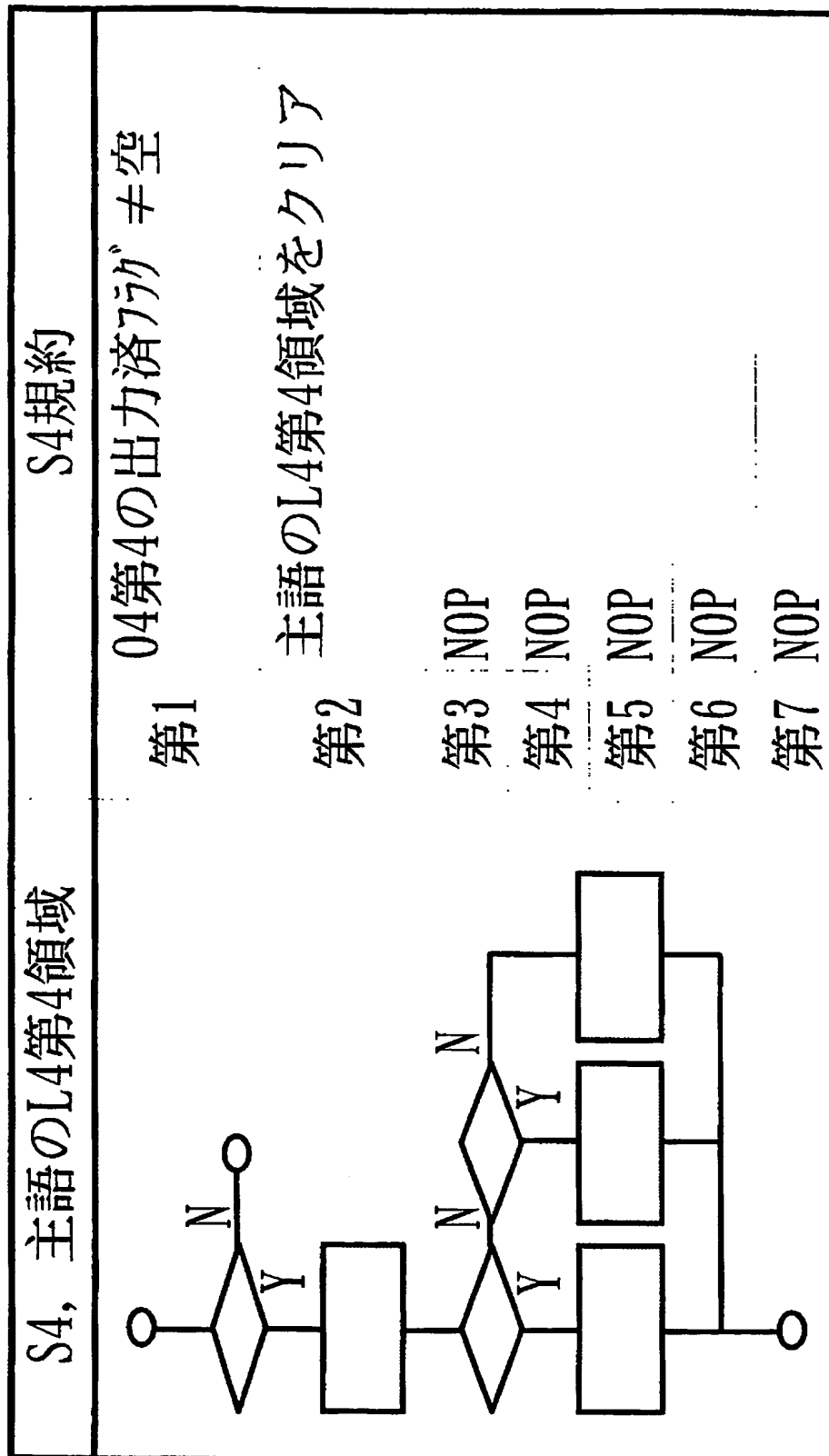
定義規則図 34：主語が出力アクセスキーのL3第4領域のS4のベクトル

【図 96】



定義規則図 35：主語が出力処理条件キーの L3 第4領域の S4 のベクトル

【図 97】



定義規則図 36: 主語が L4 第4領域の S4 のベクトル  
 定義規則 37 から 47 については定義規則図 36 参照



## 【図 98】

付録—全ベクトルの型: 001-1

正規/入力/配列無/等価無/境界無のL2 (管理番号0010-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N2
  Private W@%30@_%31@_%28@_%29@_%02@ As @%23@
Rem PUBIN2
  Private @%31@_%28@_%29@_%02@ As @%23@
Rem PUB5C2
  Private CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y2
  Private CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B2
  Private CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F2
  Private CTRL_W@%30@_%31@_%28@_%29@_%02@_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T2
  Private W@%30@_%31@_%28@_%29@_%02@_wk As @%23@
Rem PRV7R2
  Private CTRL_W@%30@_%29@_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_@%23@ As @%23@
Rem $PRV2W2
@%152@
Rem $PRV2E2
Rem $PRV4W2
@%155@
Rem $PRV4E2
Rem #DEFEND
  Public Sub Main0
Rem #VECREP Private Sub L2_@%31@_%28@_%29@_%02@()
  BOX_1:
Rem PRVLG
    If W@%30@_%31@_%28@_%29@_%02@ = CNS_NOT_KUH_@%23@ Then
      GoTo BOX_2
    End If
    GoTo BOX_E
  BOX_2:
Rem PRVLG
    @%15@
  BOX_3:
Rem PRVLG
    If W@%30@_%31@_%28@_%29@_%02@_wk <> CNS_NOT_KUH_@%23@ Then
      GoTo BOX_4
    End If
    GoTo BOX_5
  BOX_4:
Rem PRVLG
    W@%30@_%31@_%28@_%29@_%02@ = W@%30@_%31@_%28@_%29@_%02@_wk
Rem PRVLG
    CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG = CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG + 1
    GoTo BOX_E
  BOX_5:
```

## 【図 99】

付録一全ベクトルの型: 001-2

正規/入力/配列無/等価無/境界無のL2 (管理番号0010-2)

```
Rem PRVLG
  If CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P
    = CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W0%300_0%290_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 100】

付録—全ベクトルの型: 002-1

正規/出力/配列無/等価無/境界無のL3 (管理番号0020-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
Private W%300_@%310_@%280_@%290_@%020 As Integer
Rem PUBIN3
Private W%3050_@%3150_@%2850_@%2950_@%2000 As @%2350
Rem PUB5C3
Private CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_W%300_@%310_@%280_@%290_@%020_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
Private W%300_@%310_@%280_@%290_@%020_wk As Integer
Rem PRV7R3
Private CTRL_W%300_@%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_Integer As Integer
Rem $PRV2W3
@%1620
Rem $PRV2E3
Rem $PRV4W3
@%1650
Rem $PRV4E3
Rem #DEFEND
Public Sub Main()
Rem #VECREP Private Sub L3_@%310_@%280_@%290_@%020()
BOX_1:
Rem PRVLG
If W%300_@%310_@%280_@%290_@%020 = CNS_NOT_KUH_Integer Then
GoTo BOX_2
End If
GoTo BOX_E
BOX_2:
Rem PRVLG
@%160
BOX_3:
Rem PRVLG
If W%300_@%310_@%280_@%290_@%020_wk <> CNS_NOT_KUH_Integer Then
GoTo BOX_4
End If
GoTo BOX_5
BOX_4:
Rem PRVLG
W%300_@%310_@%280_@%290_@%020 = W%300_@%310_@%280_@%290_@%020_wk
Rem PRVLG
CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG = CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG + 1
GoTo BOX_E
BOX_5:
```

## 【図 1 0 1】

付録－全ベクトルの型： 0 0 2 - 2

正規／出力／配列無／等価無／境界無の L 3 (管理番号 0 0 2 0 - 2)

```
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 1 0 2】

付録－全ベクトルの型: 0 0 3 - 1

正規／出力／配列無／等価無／境界無の L 4 (管理番号 0 0 3 0 - 1)

```

Option Explicit
Rem #DEFPUB
Rem PUB4N4
  Private W0%300_0%310_0%280_0%290_0%020 As 0%230
Rem PUBIN4
  Private W0%3050_0%3150_0%2850_0%2950_0%2000 As 0%2350
Rem PUB5C4
  Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
  Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
  Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
  Private CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem PUB4B4
  Private W0%3000_0%3100_0%2800_0%2900_0%2000 As 0%230
Rem PUB4N3
  Private W3_0%310_0%280_0%290_0%020 As Integer
Rem PUB4N2
  Private W2_0%310_0%280_0%290_0%020 As 0%230
Rem #DEFPRV
Rem PRV2T4
  Private W0%300_0%310_0%280_0%290_0%020_wk As 0%230
Rem PRV7R4
  Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem *PUBKU
  Private CNS_NOT_KUH_String As String
  Private CNS_NOT_KUH_Integer As Integer
  Private CNS_NOT_KUH_Boolean As Boolean
  Private CNS_NOT_KUH_Long As Long
  Private CNS_NOT_KUH_Single As Single
  Private CNS_NOT_KUH_Double As Double
  Private CNS_NOT_KUH_Variant As Variant
  Private CNS_NOT_KUH_Currency As Currency
  Private CNS_NOT_KUH_Byte As Byte
  Private CNS_NOT_KUH_Date As Date
Rem $PRV2W4
0%1720
Rem $PRV2E4
Rem $PRV4W4
0%1820
Rem $PRV4E4
Rem #DEFEND
  Public Sub Main 0
Rem #VECREP Private Sub L4_0%310_0%280_0%290_0%020_0%070 0
  BOX_1:
Rem PRVLG
  If W0%300_0%310_0%280_0%290_0%020 = CNS_NOT_KUH_0%230 And _
Rem PRVLG
  W3_0%310_0%280_0%290_0%020 = 0%070 Then
  GoTo BOX_2
  End If

```

## 【図 103】

付録-全ベクトルの型: 003-2

正規/出力/配列無/等価無/境界無の L 4 (管理番号 0030-2)

```
GoTo BOX_E
BOX_2:
Rem PUBIN
  If W%305@_@%315@_@%285@_@%295@_@%200@ = CNS_NOT_KUH_@%235@ Then Go To Box_3
Rem PRVLG
  @%17@
BOX_3:
Rem PRVLG
  If W%30@_@%31@_@%28@_@%29@_@%02@_wk <> CNS_NOT_KUH_@%23@ Then
    GoTo BOX_4
  End If
  GoTo BOX_5
BOX_4:
Rem PRVLG
  W%30@_@%31@_@%28@_@%29@_@%02@ = W%30@_@%31@_@%28@_@%29@_@%02@_wk
Rem PRVLG
  CTRL_W%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_W%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 1 0 4】

付録一全ベクトルの型: 0 0 4 - 1

正規/入力/配列有/等価無/境界無の L 2 (管理番号 0 0 4 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N2
Private W@%30@_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As @%23@
Rem PUB1N2
Private @%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As @%23@
Rem PUB5C2
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y2
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B2
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F2
Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (@%26@, @%27@) As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T2
Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk (@%26@, @%27@) As @%23@
Rem PRV7R2
Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_@%23@ As @%23@
Private Gyou as Integer
Private Retu as Integer
Private Uti_Cnt as Integer
Private Soto_Cnt as Integer
Private Uti_Max as Integer
Private Soto_Max as Integer
Rem $PRV2W2
@%152@
Rem $PRV2E2
Rem $PRV4W2
@%155@
Rem $PRV4E2
Rem #DEFEND
Public Sub Main ()
Rem #VECREP Private Sub L2_@%31@_@%28@_@%29@_@%02@ ()
    Gyou = 0
    Retu = 0
Rem PRVLG
    If @%34@ < @%35@ then
Rem PRVLG
        Uti_Max = @%26@
Rem PRVLG
        Soto_Max = @%27@
    End If
Rem PRVLG
    If @%34@ > @%35@ then
Rem PRVLG
        Uti_Max = @%27@
Rem PRVLG
        Soto_Max = @%26@
    End If
```

## 【図 105】

付録-全ベクトルの型: 004-2

正規/入力/配列有/等価無/境界無のL2 (管理番号0040-2)

```
    Uti_Cnt = 0
    Soto_Cnt = 0
    Do Until Soto_Cnt = Soto_Max + 1
        Uti_Cnt = 0
    Rem PRVLG
        If @@34@ < @@35@ then
            Gyou = Uti_Cnt
            Retu = Soto_Cnt
        End If
    Rem PRVLG
        If @@34@ > @@35@ then
            Gyou = Soto_Cnt
            Retu = Uti_Cnt
        End If
        Do Until Uti_Cnt = Uti_Max + 1
    BOX_1:
    Rem PRVLG
        If W@@30@_@@31@_@@28@_@@29@_@@02@ (Gyou, Retu) = CNS_NOT_KUH_@@23@ Then
            GoTo BOX_2
        End If
        GoTo BOX_E
    BOX_2:
    Rem PRVLG
        @@15@
    BOX_3:
    Rem PRVLG
        If W@@30@_@@31@_@@28@_@@29@_@@02@_wk (Gyou, Retu) <> CNS_NOT_KUH_@@23@ Then
            GoTo BOX_4
        End If
        GoTo BOX_5
    BOX_4:
    Rem PRVLG
        W@@30@_@@31@_@@28@_@@29@_@@02@ (Gyou, Retu) = W@@30@_@@31@_@@28@_@@29@_@@02@_wk (Gyou, Retu)
    Rem PRVLG
        CTRL_W@@30@_@@28@_@@29@_STS_TRANSITION_FLG = CTRL_W@@30@_@@28@_@@29@_STS_TRANSITION_FLG + 1
        GoTo BOX_E
    BOX_5:
    Rem PRVLG
        If CTRL_W@@30@_@@28@_@@29@_STS_TRANSITION_FLG_P
            = CTRL_W@@30@_@@28@_@@29@_STS_TRANSITION_FLG_PP Then
            GoTo BOX_6
        End If
        GoTo BOX_7
    BOX_6:
    Rem PRVLG
        CTRL_W@@30@_@@31@_@@28@_@@29@_@@02@_NOT_VALID_FLG (Gyou, Retu) = True
        GoTo BOX_E
    BOX_7:
    Rem PRVLG
        CTRL_W@@30@_@@29@_ING_FLG = True
        GoTo BOX_E
    BOX_E:
        Uti_Cnt = Uti_Cnt + 1
```



## 【図 106】

付録－全ベクトルの型: 004-3

正規／入力／配列有／等価無／境界無のL2 (管理番号 0040-3)

---

```
Rem PRVLG
    If @%34@ < @%35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If @%34@ > @%35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
        Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
    If @%34@ < @%35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If @%34@ > @%35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
        Gyou = 0
        Retu = 0
        Uti_Cnt = 0
        Soto_Cnt = 0
        Uti_Max = 0
        Soto_Max = 0
End Sub
Rem *** PRCEND
```

---

## 【図 1 0 7】

付録一全ベクトルの型: 0 0 5 - 1

正規/出力/配列有/等価無/境界無の L 3 (管理番号 0 0 5 0 - 1)

```

Option Explicit
Rem #DEFPUB
Rem PUB4N3
Private W@%30@_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As Integer
Rem PUBIN3
Private W@%305@_@%315@_@%285@_@%295@_@%200@ (@%265@, @%275@) As @%235@
Rem PUB5C3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (@%26@, @%27@) As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk (@%26@, @%27@) As Integer
Rem PRV7R3
Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_Integer As Integer
Private Gyou as Integer
Private Retu as Integer
Private Uti_Cnt as Integer
Private Soto_Cnt as Integer
Private Uti_Max as Integer
Private Soto_Max as Integer
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub L3_@%31@_@%28@_@%29@_@%02@ ()
    Gyou = 0
    Retu = 0
Rem PRVLG
    If @%34@ < @%35@ then
Rem PRVLG
        Uti_Max = @%26@
Rem PRVLG
        Soto_Max = @%27@
    End If
Rem PRVLG
    If @%34@ > @%35@ then
Rem PRVLG
        Uti_Max = @%27@
Rem PRVLG
        Soto_Max = @%26@
    End If

```

## 【図 108】

付録-全ベクトルの型: 005-2

正規/出力/配列有/等価無/境界無のL3 (管理番号0050-2)

```

    Uti_Cnt = 0
    Soto_Cnt = 0
    Do Until Soto_Cnt = Soto_Max + 1
        Uti_Cnt = 0
    Rem PRVLG
        If @%34@ < @%35@ then
            Gyou = Uti_Cnt
            Retu = Soto_Cnt
        End If
    Rem PRVLG
        If @%34@ > @%35@ then
            Gyou = Soto_Cnt
            Retu = Uti_Cnt
        End If
        Do Until Uti_Cnt = Uti_Max + 1
    BOX_1:
    Rem PRVLG
        If W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = CNS_NOT_KUH_Integer Then
            GoTo BOX_2
        End If
        GoTo BOX_E
    BOX_2:
    Rem PRVLG
        @%16@
    BOX_3:
    Rem PRVLG
        If W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu) <> CNS_NOT_KUH_Integer Then
            GoTo BOX_4
        End If
        GoTo BOX_5
    BOX_4:
    Rem PRVLG
        W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu)
    Rem PRVLG
        CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
        GoTo BOX_E
    BOX_5:
    Rem PRVLG
        If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
            = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
            GoTo BOX_6
        End If
        GoTo BOX_7
    BOX_6:
    Rem PRVLG
        CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (Gyou, Retu) = True
        GoTo BOX_E
    BOX_7:
    Rem PRVLG
        CTRL_W@%30@_@%29@_ING_FLG = True
        GoTo BOX_E
    BOX_E:
        Uti_Cnt = Uti_Cnt + 1

```

## 【図109】

付録—全ベクトルの型: 005-3

正規/出力/配列有/等価無/境界無のL3 (管理番号0050-3)

---

```
Rem PRVLG
    If @@34@ < @@35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If @@34@ > @@35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
    Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
    If @@34@ < @@35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If @@34@ > @@35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
Gyou = 0
Retu = 0
Uti_Cnt = 0
Soto_Cnt = 0
Uti_Max = 0
Soto_Max = 0
End Sub
Rem *** PRCEND
```

---

## 【図 110】

付録—全ベクトルの型: 006-1

正規/出力/配列有/等価無/境界無のL4 (管理番号0060-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W0%300_0%310_0%280_0%290_0%020 (@%260, 0%270) As 0%230
Rem PUBIN4
Private W0%3050_0%3150_0%2850_0%2950_0%2000 (@%2650, 0%2750) As 0%2350
Rem PUB5C4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
Private CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG (@%260, 0%270) As Boolean
Rem #DEFEND
Rem PUB4B4
Private W0%3000_0%3100_0%2800_0%2900_0%2000 (@%260, 0%270) As 0%230
Rem PUB4N3
Private W3_0%310_0%280_0%290_0%020 (@%260, 0%270) As Integer
Rem PUB4N2
Private W2_0%310_0%280_0%290_0%020 (@%260, 0%270) As 0%230
Rem #DEFPRV
Rem PRV2T4
Private W0%300_0%310_0%280_0%290_0%020_wk (@%260, 0%270) As 0%230
Rem PRV7R4
Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem #PUBKU
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Private Gyou as Integer
Private Retu as Integer
Private Uti_Cnt as Integer
Private Soto_Cnt as Integer
Private Uti_Max as Integer
Private Soto_Max as Integer
Rem $PRV2W4
0%1720
Rem $PRV2E4
Rem $PRV4W4
0%1820
Rem $PRV4E4
Rem #DEFEND
Public Sub Main()
Rem #VECREP Private Sub L4_0%310_0%280_0%290_0%020_0%070()
Gyou = 0
```

## 【図 111】

付録—全ベクトルの型: 006-2

正規/出力/配列有/等価無/境界無のL4 (管理番号0060-2)

```
Retu = 0
Rem PRVLG
  If @@34@ < @@35@ then
Rem PRVLG
  Uti_Max = @@26@
Rem PRVLG
  Soto_Max = @@27@
  End If
Rem PRVLG
  If @@34@ > @@35@ then
Rem PRVLG
  Uti_Max = @@27@
Rem PRVLG
  Soto_Max = @@26@
  End If
  Uti_Cnt = 0
  Soto_Cnt = 0
  Do Until Soto_Cnt = Soto_Max + 1
    Uti_Cnt = 0
Rem PRVLG
    If @@34@ < @@35@ then
      Gyou = Uti_Cnt
      Retu = Soto_Cnt
    End If
Rem PRVLG
    If @@34@ > @@35@ then
      Gyou = Soto_Cnt
      Retu = Uti_Cnt
    End If
    Do Until Uti_Cnt = Uti_Max + 1
BOX_1:
Rem PRVLG
  If W@@30@_@@31@_@@28@_@@29@_@@02@ (Gyou, Retu) = CNS_NOT_KUH_@@23@ And _
Rem PRVLG
  W3_@@31@_@@28@_@@29@_@@02@ (Gyou, Retu) = @@07@ Then
    GoTo BOX_2
  End If
  GoTo BOX_E
BOX_2:
Rem PUBIN
  If W@@305@_@@315@_@@285@_@@295@_@@200@ (Gyou, Retu) = CNS_NOT_KUH_@@235@ Then Go To Box_3
Rem PRVLG
  @@17@
BOX_3:
Rem PRVLG
  If W@@30@_@@31@_@@28@_@@29@_@@02@_wk (Gyou, Retu) <> CNS_NOT_KUH_@@23@ Then
    GoTo BOX_4
  End If
  GoTo BOX_5
BOX_4:
Rem PRVLG
  W@@30@_@@31@_@@28@_@@29@_@@02@ (Gyou, Retu) = W@@30@_@@31@_@@28@_@@29@_@@02@_wk (Gyou, Retu)
Rem PRVLG
```

## 【図 112】

付録-全ベクトルの型: 006-3

正規/出力/配列有/等価無/境界無のL4 (管理番号0060-3)

```

CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG = CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG + 1
GoTo BOX_E
BOX_5:
Rem PRVLG
If CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG_P
= CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG_PP Then
GoTo BOX_6
End If
GoTo BOX_7
BOX_6:
Rem PRVLG
CTRL_W@%30@_%31@_%28@_%29@_%02@_NOT_VALID_FLG (Gyou, Retu) = True
GoTo BOX_E
BOX_7:
Rem PRVLG
CTRL_W@%30@_%29@_ING_FLG = True
GoTo BOX_E
BOX_E:
Uti_Cnt = Uti_Cnt + 1
Rem PRVLG
If @%34@ < @%35@ then
Gyou = Uti_Cnt
Retu = Soto_Cnt
End If
Rem PRVLG
If @%34@ > @%35@ then
Gyou = Soto_Cnt
Retu = Uti_Cnt
End If
Loop
Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
If @%34@ < @%35@ then
Gyou = Uti_Cnt
Retu = Soto_Cnt
End If
Rem PRVLG
If @%34@ > @%35@ then
Gyou = Soto_Cnt
Retu = Uti_Cnt
End If
Loop
Gyou = 0
Retu = 0
Uti_Cnt = 0
Soto_Cnt = 0
Uti_Max = 0
Soto_Max = 0
End Sub
Rem *** PRCEND

```

## 【図 1 1 3】

付録－全ベクトルの型: 0 0 7 - 1

正規／出力／配列無／等価有／境界無の L 3 (管理番号 0 0 7 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
Private W@%30@_@%31@_@%28@_@%29@_@%02@ As Integer
Rem PUBIN3
Private W@%305@_@%315@_@%285@_@%295@_@%200@ As @%235@
Rem PUB5C3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ As Integer
Rem PRV7R3
Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_Integer As Integer
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
Public Sub Main 0
Rem #VECREP Private Sub L3_@%31@_@%28@_@%29@_@%02@ ()
BOX_1:
Rem PRVLG
If W@%30@_@%31@_@%28@_@%29@_@%02@ = CNS_NOT_KUH_Integer Then
GoTo BOX_2
End If
GoTo BOX_E
BOX_2:
Rem PRVLG
@%16@
BOX_3:
Rem PUBL3F
If W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ = 1 Then GoTo BOX_4
GoTo BOX_5
BOX_4:
Rem PUBL3F
If W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ = 1 Then W@%30@_@%31@_@%28@_@%29@_@%02@ = @%07@
Rem PRVLG
CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
GoTo BOX_E
BOX_5:
Rem PRVLG
If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
```



## 【図 1 1 4】

付録-全ベクトルの型: 007-2

正規/出力/配列無/等価有/境界無のL3 (管理番号0070-2)

```
= CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP Then
  GoTo BOX_6
End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W0%300_0%290_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 115】

付録-全ベクトルの型: 008-1

正規/出力/配列無/等価有/境界無のL4 (管理番号 0080-1)

```

Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W0%300_@%310_@%280_@%290_@%020 As @%230
Rem PUBIN4
Private W0%3050_@%3150_@%2850_@%2950_@%2000 As @%2350
Rem PUB5C4
Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
Private CTRL_W0%300_@%310_@%280_@%290_@%020_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem PUB4B4
Private W0%3000_@%3100_@%2800_@%2900_@%2000 As @%230
Rem PUB4N3
Private W3_@%310_@%280_@%290_@%020 As Integer
Rem PUB4N2
' Private W2_@%310_@%280_@%290_@%020 As @%230
Rem #DEFPRV
Rem PRV2T4
Private W0%300_@%310_@%280_@%290_@%020_wk As @%230
Rem PRV7R4
Private CTRL_W0%300_@%290_ING_FLG As Boolean
Rem *PUBKU
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Rem $PRV2W4
@%1720
Rem $PRV2E4
Rem $PRV4W4
@%1820
Rem $PRV4E4
Rem #DEFEND
Public Sub Main 0
Rem #VECREP Private Sub L4_@%310_@%280_@%290_@%020_@%070 0
BOX_1:
Rem PRVLG
If W0%300_@%310_@%280_@%290_@%020 = CNS_NOT_KUH_@%230 And _
Rem PRVLG
W3_@%310_@%280_@%290_@%020 = @%070 Then
GoTo BOX_2
End If

```

## 【図 116】

付録－全ベクトルの型：008-2

正規／出力／配列無／等価有／境界無のL4（管理番号0080-2）

```
GoTo BOX_E
BOX_2:
Rem PUBIN
  If W@%305@_@%315@_@%285@_@%295@_@%200@ = CNS_NOT_KUH_@%235@ Then Go To Box_3
Rem PRVLG
  @%17@
BOX_3:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@_wk <> CNS_NOT_KUH_@%23@ Then
    GoTo BOX_4
  End If
  GoTo BOX_5
BOX_4:
Rem PRVLG
  W@%30@_@%31@_@%28@_@%29@_@%02@ = W@%30@_@%31@_@%28@_@%29@_@%02@_wk
Rem PRVLG
  CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 117】

付録—全ベクトルの型: 009-1

正規/出力/配列有/等価有/境界無のL3 (管理番号0090-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
Private W@%30@_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As Integer
Rem PUBIN3
Private W@%305@_@%315@_@%285@_@%295@_@%200@ (@%265@, @%275@) As @%235@
Rem PUB5C3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (@%26@, @%27@) As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ (@%26@, @%27@) As Integer
Rem PRV7R3
Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_Integer As Integer
Private Gyou as Integer
Private Retu as Integer
Private Uti_Cnt as Integer
Private Soto_Cnt as Integer
Private Uti_Max as Integer
Private Soto_Max as Integer
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
Public Sub Main()
Rem #VECREP Private Sub L3_@%31@_@%28@_@%29@_@%02@ ()
    Gyou = 0
    Retu = 0
Rem PRVLG
    If @%34@ < @%35@ then
Rem PRVLG
        Uti_Max = @%26@
Rem PRVLG
        Soto_Max = @%27@
    End If
Rem PRVLG
    If @%34@ > @%35@ then
Rem PRVLG
        Uti_Max = @%27@
Rem PRVLG
        Soto_Max = @%26@
    End If
```

## 【図 118】

付録一全ベクトルの型: 009-2

正規/出力/配列有/等価有/境界無のL3 (管理番号0090-2)

```
      Uti_Cnt = 0
      Soto_Cnt = 0
      Do Until Soto_Cnt = Soto_Max + 1
        Uti_Cnt = 0
      Rem PRVLG
        If @%34@ < @%35@ then
          Gyou = Uti_Cnt
          Retu = Soto_Cnt
        End If
      Rem PRVLG
        If @%34@ > @%35@ then
          Gyou = Soto_Cnt
          Retu = Uti_Cnt
        End If
      Do Until Uti_Cnt = Uti_Max + 1
      BOX_1:
      Rem PRVLG
        If W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = CNS_NOT_KUH_Integer Then
          GoTo BOX_2
        End If
        GoTo BOX_E
      BOX_2:
      Rem PRVLG
        @%16@
      BOX_3:
      Rem PUBL3F
        If W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ (Gyou, Retu) = 1 Then GoTo BOX_4
        GoTo BOX_5
      BOX_4:
      Rem PUBL3F
        If W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ (Gyou, Retu) = 1
          Then W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = @%07@
      Rem PRVLG
        CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
        GoTo BOX_E
      BOX_5:
      Rem PRVLG
        If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
          = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
          GoTo BOX_6
        End If
        GoTo BOX_7
      BOX_6:
      Rem PRVLG
        CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (Gyou, Retu) = True
        GoTo BOX_E
      BOX_7:
      Rem PRVLG
        CTRL_W@%30@_@%29@_ING_FLG = True
        GoTo BOX_E
      BOX_E:
        Uti_Cnt = Uti_Cnt + 1
      Rem PRVLG
```

## 【図 1 1 9】

付録－全ベクトルの型：0 0 9－3

正規／出力／配列有／等価有／境界無の L 3（管理番号 0 0 9 0－3）

```
      If @%34@ < @%35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
      End If
Rem PRVLG
      If @%34@ > @%35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
      End If
      Loop
      Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
      If @%34@ < @%35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
      End If
Rem PRVLG
      If @%34@ > @%35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
      End If
      Loop
      Gyou = 0
      Retu = 0
      Uti_Cnt = 0
      Soto_Cnt = 0
      Uti_Max = 0
      Soto_Max = 0
End Sub
Rem *** PRCEND
```

## 【図 1 2 0】

付録－全ベクトルの型: 0 1 0 - 1

正規／出力／配列有／等価有／境界無の L 4 (管理番号 0 1 0 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
  Private W@%30@_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As @%23@
Rem PUBIN4
  Private W@%305@_@%315@_@%285@_@%295@_@%200@ (@%265@, @%275@) As @%235@
Rem PUB5C4
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
  Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (@%26@, @%27@) As Boolean
Rem #DEFEND
Rem PUB4B4
  Private W@%300@_@%310@_@%280@_@%290@_@%200@ (@%26@, @%27@) As @%23@
Rem PUB4N3
  Private W3_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As Integer
Rem PUB4N2
  Private W2_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As @%23@
Rem #DEFPRV
Rem PRV2T4
  Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk (@%26@, @%27@) As @%23@
Rem PRV7R4
  Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem *PUBKU
  Private CNS_NOT_KUH_String As String
  Private CNS_NOT_KUH_Integer As Integer
  Private CNS_NOT_KUH_Boolean As Boolean
  Private CNS_NOT_KUH_Long As Long
  Private CNS_NOT_KUH_Single As Single
  Private CNS_NOT_KUH_Double As Double
  Private CNS_NOT_KUH_Variant As Variant
  Private CNS_NOT_KUH_Currency As Currency
  Private CNS_NOT_KUH_Byte As Byte
  Private CNS_NOT_KUH_Date As Date
  Private Gyou as Integer
  Private Retu as Integer
  Private Uti_Cnt as Integer
  Private Soto_Cnt as Integer
  Private Uti_Max as Integer
  Private Soto_Max as Integer
Rem $PRV2W4
@%172@
Rem $PRV2E4
Rem $PRV4W4
@%182@
Rem $PRV4E4
Rem #DEFEND
  Public Sub Main ()
Rem #VECREP Private Sub L4_@%31@_@%28@_@%29@_@%02@_@%07@ ()
  Gyou = 0
```

## 【図 1 2 1】

付録一全ベクトルの型: 0 1 0 - 2

正規/出力/配列有/等価有/境界無の L 4 (管理番号 0 1 0 0 - 2)

```
Retu = 0
Rem PRVLG
  If @%34@ < @%35@ then
Rem PRVLG
  Uti_Max = @%26@
Rem PRVLG
  Soto_Max = @%27@
  End If
Rem PRVLG
  If @%34@ > @%35@ then
Rem PRVLG
  Uti_Max = @%27@
Rem PRVLG
  Soto_Max = @%26@
  End If
  Uti_Cnt = 0
  Soto_Cnt = 0
  Do Until Soto_Cnt = Soto_Max + 1
    Uti_Cnt = 0
Rem PRVLG
    If @%34@ < @%35@ then
      Gyou = Uti_Cnt
      Retu = Soto_Cnt
    End If
Rem PRVLG
    If @%34@ > @%35@ then
      Gyou = Soto_Cnt
      Retu = Uti_Cnt
    End If
    Do Until Uti_Cnt = Uti_Max + 1
BOX_1:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = CNS_NOT_KUH_@%23@ And _
Rem PRVLG
  W3_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = @%07@ Then
    GoTo BOX_2
  End If
  GoTo BOX_E
BOX_2:
Rem PUBIN
  If W@%305@_@%315@_@%285@_@%295@_@%200@ (Gyou, Retu) = CNS_NOT_KUH_@%235@ Then Go To Box_3
Rem PRVLG
  @%17@
BOX_3:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu) <> CNS_NOT_KUH_@%23@ Then
    GoTo BOX_4
  End If
  GoTo BOX_5
BOX_4:
Rem PRVLG
  W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu)
Rem PRVLG
```



## 【図 1 2 2】

付録-全ベクトルの型: 010-3

正規/出力/配列有/等価有/境界無のL4 (管理番号0100-3)

```
CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
GoTo BOX_E
BOX_5:
Rem PRVLG
If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
= CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
GoTo BOX_6
End If
GoTo BOX_7
BOX_6:
Rem PRVLG
CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (Gyou, Retu) = True
GoTo BOX_E
BOX_7:
Rem PRVLG
CTRL_W@%30@_@%29@_ING_FLG = True
GoTo BOX_E
BOX_E:
Uti_Cnt = Uti_Cnt + 1
Rem PRVLG
If @%34@ < @%35@ then
Gyou = Uti_Cnt
Retu = Soto_Cnt
End If
Rem PRVLG
If @%34@ > @%35@ then
Gyou = Soto_Cnt
Retu = Uti_Cnt
End If
Loop
Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
If @%34@ < @%35@ then
Gyou = Uti_Cnt
Retu = Soto_Cnt
End If
Rem PRVLG
If @%34@ > @%35@ then
Gyou = Soto_Cnt
Retu = Uti_Cnt
End If
Loop
Gyou = 0
Retu = 0
Uti_Cnt = 0
Soto_Cnt = 0
Uti_Max = 0
Soto_Max = 0
End Sub
Rem *** PRCEND
```

## 【図 1 2 3】

付録—全ベクトルの型: 0 1 1 - 1

正規/出力/配列無/等価無/境界有の L 3 (管理番号 0 1 1 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
  Private W@%30@_@%31@_@%28@_@%29@_@%02@ As Integer
Rem PUBIN3
  Private W@%305@_@%315@_@%285@_@%295@_@%200@ As @%235@
Rem PUB5C3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
  Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
  Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk As Integer
Rem PRV7R3
  Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_Integer As Integer
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
  Public Sub Main0
Rem #VECREP Private Sub L3_@%31@_@%28@_@%29@_@%02@()
  BOX_1:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@ = CNS_NOT_KUH_Integer Then
    GoTo BOX_2
  End If
  GoTo BOX_E
  BOX_2:
Rem PRVLG
  @%16@
  BOX_3:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@_wk <> CNS_NOT_KUH_Integer Then
    GoTo BOX_4
  End If
  GoTo BOX_5
  BOX_4:
Rem PRVLG
  W@%30@_@%31@_@%28@_@%29@_@%02@ = W@%30@_@%31@_@%28@_@%29@_@%02@_wk
Rem PRVLG
  CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
  BOX_5:
```

## 【図 1 2 4】

付録－全ベクトルの型: 0 1 1 - 2

正規/出力/配列無/等価無/境界有の L 3 (管理番号 0 1 1 0 - 2)

```
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 1 2 5】

付録-全ベクトルの型: 0 1 2 - 1

正規/出力/配列無/等価無/境界有の L 4 (管理番号 0 1 2 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
  Private W@%30@_@%31@_@%28@_@%29@_@%02@ As @%23@
Rem PUBIN4
  Private W@%305@_@%315@_@%285@_@%295@_@%200@ As @%235@
Rem PUB5C4
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
  Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem PUB4B4
  Private W@%300@_@%310@_@%280@_@%290@_@%200@ As @%23@
Rem PUB4N3
  Private W3_@%31@_@%28@_@%29@_@%02@ As Integer
Rem PUB4N2
  Private W2_@%31@_@%28@_@%29@_@%02@ As @%23@
Rem #DEFPRV
Rem PRV2T4
  Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk As @%23@
Rem PRV7R4
  Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem *PUBKU
  Private CNS_NOT_KUH_String As String
  Private CNS_NOT_KUH_Integer As Integer
  Private CNS_NOT_KUH_Boolean As Boolean
  Private CNS_NOT_KUH_Long As Long
  Private CNS_NOT_KUH_Single As Single
  Private CNS_NOT_KUH_Double As Double
  Private CNS_NOT_KUH_Variant As Variant
  Private CNS_NOT_KUH_Currency As Currency
  Private CNS_NOT_KUH_Byte As Byte
  Private CNS_NOT_KUH_Date As Date
Rem $PRV2W4
@%172@
Rem $PRV2E4
Rem $PRV4W4
@%182@
Rem $PRV4E4
Rem #DEFEND
  Public Sub Main()
Rem #VECREP Private Sub L4_@%31@_@%28@_@%29@_@%02@_@%07@()
  BOX_1:
Rem PRVLG
    If W@%30@_@%31@_@%28@_@%29@_@%02@ = CNS_NOT_KUH_@%23@ And _
Rem PRVLG
    W3_@%31@_@%28@_@%29@_@%02@ = @%07@ Then
      GoTo BOX_2
    End If
```

## 【図 1 2 6】

付録-全ベクトルの型: 0 1 2 - 2

正規/出力/配列無/等価無/境界有の L 4 (管理番号 0 1 2 0 - 2)

```
GoTo BOX_E
BOX_2:
Rem PUBIN
  If W@%305@_@%315@_@%285@_@%295@_@%200@ = CNS_NOT_KUH_@%235@ Then Go To Box_3
Rem PRVLG
  @%17@
BOX_3:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@_wk <> CNS_NOT_KUH_@%23@ Then
    GoTo BOX_4
  End If
  GoTo BOX_5
BOX_4:
Rem PRVLG
  W@%30@_@%31@_@%28@_@%29@_@%02@ = W@%30@_@%31@_@%28@_@%29@_@%02@_wk
Rem PRVLG
  W@%300@_@%310@_@%280@_@%290@_@%200@ = W@%30@_@%31@_@%28@_@%29@_@%02@
Rem PRVLG
  CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 1 2 7】

付録—全ベクトルの型: 0 1 3 - 1

正規/出力/配列無/等価有/境界有の L 3 (管理番号 0 1 3 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
  Private W@%30@_@%31@_@%28@_@%29@_@%02@ As Integer
Rem PUBIN3
  Private W@%305@_@%315@_@%285@_@%295@_@%200@ As @%235@
Rem PUB5C3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
  Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
  Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ As Integer
Rem PRV7R3
  Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_Integer As Integer
Rem $PRV2W3
  @%162@
Rem $PRV2E3
Rem $PRV4W3
  @%165@
Rem $PRV4E3
Rem #DEFEND
  Public Sub Main()
Rem #VECREP Private Sub L3_@%31@_@%28@_@%29@_@%02@()
  BOX_1:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@ = CNS_NOT_KUH_Integer Then
    GoTo BOX_2
  End If
  GoTo BOX_E
  BOX_2:
Rem PRVLG
  @%16@
  BOX_3:
Rem PUBL3F
  If W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ = 1 Then GoTo BOX_4
  GoTo BOX_5
  BOX_4:
Rem PUBL3F
  If W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ = 1 Then W@%30@_@%31@_@%28@_@%29@_@%02@ = @%07@
Rem PRVLG
  CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
  BOX_5:
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
```

## 【図 128】

付録一全ベクトルの型: 013-2

正規/出力/配列無/等価有/境界有のL3 (管理番号0130-2)

```
= CTRL_W0%30@_0%28@_0%29@_STS_TRANSITION_FLG_PP Then
  GoTo BOX_6
End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W0%30@_0%31@_0%28@_0%29@_0%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W0%30@_0%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 129】

付録-全ベクトルの型: 014-1

正規/出力/配列無/等価有/境界有のL4 (管理番号0140-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
  Private W0%300_0%310_0%280_0%290_0%020 As 0%230
Rem PUBIN4
  Private W0%3050_0%3150_0%2850_0%2950_0%2000 As 0%2350
Rem PUB5C4
  Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
  Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
  Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
  Private CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem PUB4B4
  Private W0%3000_0%3100_0%2800_0%2900_0%2000 As 0%230
Rem PUB4N3
  Private W3_0%310_0%280_0%290_0%020 As Integer
Rem PUB4N2
  ' Private W2_0%310_0%280_0%290_0%020 As 0%230
Rem #DEFPRV
Rem PRV2T4
  Private W0%300_0%310_0%280_0%290_0%020_wk As 0%230
Rem PRV7R4
  Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem *PUBKU
  Private CNS_NOT_KUH_String As String
  Private CNS_NOT_KUH_Integer As Integer
  Private CNS_NOT_KUH_Boolean As Boolean
  Private CNS_NOT_KUH_Long As Long
  Private CNS_NOT_KUH_Single As Single
  Private CNS_NOT_KUH_Double As Double
  Private CNS_NOT_KUH_Variant As Variant
  Private CNS_NOT_KUH_Currency As Currency
  Private CNS_NOT_KUH_Byte As Byte
  Private CNS_NOT_KUH_Date As Date
Rem $PRV2W4
0%1720
Rem $PRV2E4
Rem $PRV4W4
0%1820
Rem $PRV4E4
Rem #DEFEND
  Public Sub Main()
Rem #VECREP Private Sub L4_0%310_0%280_0%290_0%020_0%070()
  BOX_1:
Rem PRVLG
    If W0%300_0%310_0%280_0%290_0%020 = CNS_NOT_KUH_0%230 And _
Rem PRVLG
    W3_0%310_0%280_0%290_0%020 = 0%070 Then
      GoTo BOX_2
    End If
```



## 【図 130】

付録-全ベクトルの型: 014-2

正規/出力/配列無/等価有/境界有のL4 (管理番号0140-2)

```
GoTo BOX_E
BOX_2:
Rem PUBIN
  If W@%305@_@%315@_@%285@_@%295@_@%200@ = CNS_NOT_KUH_@%235@ Then Go To Box_3
Rem PRVLG
  @%17@
BOX_3:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@_wk <> CNS_NOT_KUH_@%23@ Then
    GoTo BOX_4
  End If
  GoTo BOX_5
BOX_4:
Rem PRVLG
  W@%30@_@%31@_@%28@_@%29@_@%02@ = W@%30@_@%31@_@%28@_@%29@_@%02@_wk
Rem PRVLG
  W@%300@_@%310@_@%280@_@%290@_@%200@ = W@%30@_@%31@_@%28@_@%29@_@%02@
Rem PRVLG
  CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 131】

付録-全ベクトルの型: 015-1

正規/出力/配列有/等価無/境界有のL3 (管理番号0150-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
  Private W@%30@_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As Integer
Rem PUBIN3
  Private W@%305@_@%315@_@%285@_@%295@_@%200@ (@%265@, @%275@) As @%235@
Rem PUB5C3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
  Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (@%26@, @%27@) As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
  Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk (@%26@, @%27@) As Integer
Rem PRV7R3
  Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_Integer As Integer
  Private Gyou as Integer
  Private Retu as Integer
  Private Uti_Cnt as Integer
  Private Soto_Cnt as Integer
  Private Uti_Max as Integer
  Private Soto_Max as Integer
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
  Public Sub Main ()
Rem #VECREP Private Sub L3_@%31@_@%28@_@%29@_@%02@ ()
  Gyou = 0
  Retu = 0
Rem PRVLG
  If @%34@ < @%35@ then
Rem PRVLG
    Uti_Max = @%26@
Rem PRVLG
    Soto_Max = @%27@
  End If
Rem PRVLG
  If @%34@ > @%35@ then
Rem PRVLG
    Uti_Max = @%27@
Rem PRVLG
    Soto_Max = @%26@
  End If
```

## 【図 132】

付録—全ベクトルの型: 015-2

正規/出力/配列有/等価無/境界有のL3 (管理番号0150-2)

```

    Uti_Cnt = 0
    Soto_Cnt = 0
    Do Until Soto_Cnt = Soto_Max + 1
        Uti_Cnt = 0
    Rem PRVLG
        If @@%34@ < @@%35@ then
            Gyou = Uti_Cnt
            Retu = Soto_Cnt
        End If
    Rem PRVLG
        If @@%34@ > @@%35@ then
            Gyou = Soto_Cnt
            Retu = Uti_Cnt
        End If
        Do Until Uti_Cnt = Uti_Max + 1
    BOX_1:
    Rem PRVLG
        If W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = CNS_NOT_KUH_Integer Then
            GoTo BOX_2
        End If
        GoTo BOX_E
    BOX_2:
    Rem PRVLG
        @@%16@
    BOX_3:
    Rem PRVLG
        If W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu) <> CNS_NOT_KUH_Integer Then
            GoTo BOX_4
        End If
        GoTo BOX_5
    BOX_4:
    Rem PRVLG
        W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu)
    Rem PRVLG
        CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
        GoTo BOX_E
    BOX_5:
    Rem PRVLG
        If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
            = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
            GoTo BOX_6
        End If
        GoTo BOX_7
    BOX_6:
    Rem PRVLG
        CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (Gyou, Retu) = True
        GoTo BOX_E
    BOX_7:
    Rem PRVLG
        CTRL_W@%30@_@%29@_ING_FLG = True
        GoTo BOX_E
    BOX_E:
        Uti_Cnt = Uti_Cnt + 1

```

## 【図 133】

付録-全ベクトルの型: 015-3

正規/出力/配列有/等価無/境界有のL3 (管理番号0150-3)

```
Rem PRVLG
    If @%34@ < @%35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If @%34@ > @%35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
    Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
    If @%34@ < @%35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If @%34@ > @%35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
    Gyou = 0
    Retu = 0
    Uti_Cnt = 0
    Soto_Cnt = 0
    Uti_Max = 0
    Soto_Max = 0
End Sub
Rem *** PRCEND
```

## 【図 1 3 4】

付録一全ベクトルの型 : 0 1 6 - 1

正規／出力／配列有／等価無／境界有の L 4 (管理番号 0 1 6 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W@%30@_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As @%23@
Rem PUBIN4
Private W@%305@_@%315@_@%285@_@%295@_@%200@ (@%265@, @%275@) As @%235@
Rem PUB5C4
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (@%26@, @%27@) As Boolean
Rem #DEFEND
Rem PUB4B4
Private W@%300@_@%310@_@%280@_@%290@_@%200@ (@%26@, @%27@) As @%23@
Rem PUB4N3
Private W3_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As Integer
Rem PUB4N2
Private W2_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As @%23@
Rem #DEFPRV
Rem PRV2T4
Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk (@%26@, @%27@) As @%23@
Rem PRV7R4
Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem *PUBKU
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Private Gyou as Integer
Private Retu as Integer
Private Uti_Cnt as Integer
Private Soto_Cnt as Integer
Private Uti_Max as Integer
Private Soto_Max as Integer
Rem $PRV2W4
@%172@
Rem $PRV2E4
Rem $PRV4W4
@%182@
Rem $PRV4E4
Rem #DEFEND
Public Sub Main 0
Rem #VECREP Private Sub L4_@%31@_@%28@_@%29@_@%02@ 0
Gyou = 0
```

## 【図 1 3 5】

付録—全ベクトルの型: 0 1 6 - 2

正規/出力/配列有/等価無/境界有の L 4 (管理番号 0 1 6 0 - 2)

```
Retu = 0
Rem PRVLG
  If @%34@ < @%35@ then
Rem PRVLG
  Uti_Max = @%26@
Rem PRVLG
  Soto_Max = @%27@
  End If
Rem PRVLG
  If @%34@ > @%35@ then
Rem PRVLG
  Uti_Max = @%27@
Rem PRVLG
  Soto_Max = @%26@
  End If
  Uti_Cnt = 0
  Soto_Cnt = 0
  Do Until Soto_Cnt = Soto_Max + 1
    Uti_Cnt = 0
Rem PRVLG
    If @%34@ < @%35@ then
      Gyou = Uti_Cnt
      Retu = Soto_Cnt
    End If
Rem PRVLG
    If @%34@ > @%35@ then
      Gyou = Soto_Cnt
      Retu = Uti_Cnt
    End If
    Do Until Uti_Cnt = Uti_Max + 1
BOX_1:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = CNS_NOT_KUH_@%23@ And _
Rem PRVLG
  W3_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = @%07@ Then
    GoTo BOX_2
  End If
    GoTo BOX_E
BOX_2:
Rem PUBIN
  If W@%305@_@%315@_@%285@_@%295@_@%200@ (Gyou, Retu) = CNS_NOT_KUH_@%235@ Then Go To Box_3
Rem PRVLG
  @%17@
BOX_3:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu) <> CNS_NOT_KUH_@%23@ Then
    GoTo BOX_4
  End If
    GoTo BOX_5
BOX_4:
Rem PRVLG
  W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu)
Rem PRVLG
```

## 【図 1 3 6】

付録—全ベクトルの型: 0 1 6 - 3

正規/出力/配列有/等価無/境界有の L 4 (管理番号 0 1 6 0 - 3)

```
W0%300@_0%310@_0%280@_0%290@_0%200@ (Gyou, Retu) = W0%300@_0%310@_0%280@_0%290@_0%020@ (Gyou, Retu)
Rem PRVLG
    CTRL_W0%300@_0%280@_0%290@_STS_TRANSITION_FLG = CTRL_W0%300@_0%280@_0%290@_STS_TRANSITION_FLG + 1
    GoTo BOX_E
BOX_5:
Rem PRVLG
    If CTRL_W0%300@_0%280@_0%290@_STS_TRANSITION_FLG_P
        = CTRL_W0%300@_0%280@_0%290@_STS_TRANSITION_FLG_PP Then
        GoTo BOX_6
    End If
    GoTo BOX_7
BOX_6:
Rem PRVLG
    CTRL_W0%300@_0%310@_0%280@_0%290@_0%020@_NOT_VALID_FLG (Gyou, Retu) = True
    GoTo BOX_E
BOX_7:
Rem PRVLG
    CTRL_W0%300@_0%290@_ING_FLG = True
    GoTo BOX_E
BOX_E:
    Uti_Cnt = Uti_Cnt + 1
Rem PRVLG
    If 0%340 < 0%350 then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If 0%340 > 0%350 then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
    Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
    If 0%340 < 0%350 then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If 0%340 > 0%350 then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
    Gyou = 0
    Retu = 0
    Uti_Cnt = 0
    Soto_Cnt = 0
    Uti_Max = 0
    Soto_Max = 0
End Sub
Rem *** PRCEND
```

## 【図 1 3 7】

付録-全ベクトルの型: 0 1 7 - 1

正規/出力/配列有/等価有/境界有の L 3 (管理番号 0 1 7 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
  Private W%30@_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As Integer
Rem PUBIN3
  Private W%305@_@%315@_@%285@_@%295@_@%200@ (@%265@, @%275@) As @%235@
Rem PUB5C3
  Private CTRL_W%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
  Private CTRL_W%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
  Private CTRL_W%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
  Private CTRL_W%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (@%26@, @%27@) As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
  Private W%30@_@%31@_@%28@_@%29@_@%02@_wk%07@ (@%26@, @%27@) As Integer
Rem PRV7R3
  Private CTRL_W%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_Integer As Integer
  Private Gyou as Integer
  Private Retu as Integer
  Private Uti_Cnt as Integer
  Private Soto_Cnt as Integer
  Private Uti_Max as Integer
  Private Soto_Max as Integer
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
  Public Sub Main ()
Rem #VECREP Private Sub L3_@%31@_@%28@_@%29@_@%02@ ()
    Gyou = 0
    Retu = 0
Rem PRVLG
    If @%34@ < @%35@ then
Rem PRVLG
        Uti_Max = @%26@
Rem PRVLG
        Soto_Max = @%27@
    End If
Rem PRVLG
    If @%34@ > @%35@ then
Rem PRVLG
        Uti_Max = @%27@
Rem PRVLG
        Soto_Max = @%26@
    End If
```



## 【図 138】

付録-全ベクトルの型: 017-2

正規/出力/配列有/等価有/境界有のL3 (管理番号0170-2)

```

    Uti_Cnt = 0
    Soto_Cnt = 0
    Do Until Soto_Cnt = Soto_Max + 1
        Uti_Cnt = 0
    Rem PRVLG
        If %340 < %350 then
            Gyou = Uti_Cnt
            Retu = Soto_Cnt
        End If
    Rem PRVLG
        If %340 > %350 then
            Gyou = Soto_Cnt
            Retu = Uti_Cnt
        End If
        Do Until Uti_Cnt = Uti_Max + 1
    BOX_1:
    Rem PRVLG
        If W%300_%310_%280_%290_%020 (Gyou, Retu) = CNS_NOT_KUH_Integer Then
            GoTo BOX_2
        End If
        GoTo BOX_E
    BOX_2:
    Rem PRVLG
        %160
    BOX_3:
    Rem PUBL3F
        If W%300_%310_%280_%290_%020_wk%070 (Gyou, Retu) = 1 Then GoTo BOX_4
        GoTo BOX_5
    BOX_4:
    Rem PUBL3F
        If W%300_%310_%280_%290_%020_wk%070 (Gyou, Retu) = 1 Then W%300_%310_%280_%290_%020 (Gyou, Retu) = %0'
    Rem PRVLG
        CTRL_W%300_%280_%290_STS_TRANSITION_FLG = CTRL_W%300_%280_%290_STS_TRANSITION_FLG + 1
        GoTo BOX_E
    BOX_5:
    Rem PRVLG
        If CTRL_W%300_%280_%290_STS_TRANSITION_FLG_P
            = CTRL_W%300_%280_%290_STS_TRANSITION_FLG_PP Then
            GoTo BOX_6
        End If
        GoTo BOX_7
    Rem PRVLG
        If CTRL_W%300_%280_%290_STS_TRANSITION_FLG_P
            = CTRL_W%300_%280_%290_STS_TRANSITION_FLG_PP Then
            GoTo BOX_6
        End If
        GoTo BOX_7
    BOX_6:
    Rem PRVLG
        CTRL_W%300_%310_%280_%290_%020_NOT_VALID_FLG (Gyou, Retu) = True
        GoTo BOX_E
    BOX_7:
    Rem PRVLG

```

## 【図 1 3 9】

付録—全ベクトルの型: 0 1 7 - 3

正規/出力/配列有/等価有/境界有の L 3 (管理番号 0 1 7 0 - 3)

```
CTRL_W@%30@_@%29@_ING_FLG = True
GoTo BOX_E
BOX_E:
    Uti_Cnt = Uti_Cnt + 1
Rem PRVLG
    If @%34@ < @%35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If @%34@ > @%35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
    Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
    If @%34@ < @%35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If @%34@ > @%35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
Gyou = 0
Retu = 0
Uti_Cnt = 0
Soto_Cnt = 0
Uti_Max = 0
Soto_Max = 0
End Sub
Rem *** PRCEND
```

## 【図 140】

付録一全ベクトルの型: 018-1

正規/出力/配列有/等価有/境界有のL4 (管理番号0180-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W0%300_0%310_0%280_0%290_0%020 (0%260, 0%270) As 0%230
Rem PUBIN4
Private W0%3050_0%3150_0%2850_0%2950_0%2000 (0%2650, 0%2750) As 0%2350
Rem PUB5C4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
Private CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG (0%260, 0%270) As Boolean
Rem #DEFEND
Rem PUB4B4
Private W0%3000_0%3100_0%2800_0%2900_0%2000 (0%260, 0%270) As 0%230
Rem PUB4N3
Private W3_0%310_0%280_0%290_0%020 (0%260, 0%270) As Integer
Rem PUB4N2
Private W2_0%310_0%280_0%290_0%020 (0%260, 0%270) As 0%230
Rem #DEFPRV
Rem PRV2T4
Private W0%300_0%310_0%280_0%290_0%020_wk (0%260, 0%270) As 0%230
Rem PRV7R4
Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem #PUBKU
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Private Gyou as Integer
Private Retu as Integer
Private Uti_Cnt as Integer
Private Soto_Cnt as Integer
Private Uti_Max as Integer
Private Soto_Max as Integer
Rem $PRV2W4
0%1720
Rem $PRV2E4
Rem $PRV4W4
0%1820
Rem $PRV4E4
Rem #DEFEND
Public Sub Main()
Rem #VECREP Private Sub L4_0%310_0%280_0%290_0%020_0%070()
Gyou = 0
```

## 【図 141】

付録一全ベクトルの型: 018-2

正規/出力/配列有/等価有/境界有のL4 (管理番号0180-2)

```

Retu = 0
Rem PRVLG
  If @@34@ < @@35@ then
Rem PRVLG
  Uti_Max = @@26@
Rem PRVLG
  Soto_Max = @@27@
  End If
Rem PRVLG
  If @@34@ > @@35@ then
Rem PRVLG
  Uti_Max = @@27@
Rem PRVLG
  Soto_Max = @@26@
  End If
  Uti_Cnt = 0
  Soto_Cnt = 0
  Do Until Soto_Cnt = Soto_Max + 1
    Uti_Cnt = 0
Rem PRVLG
    If @@34@ < @@35@ then
      Gyou = Uti_Cnt
      Retu = Soto_Cnt
    End If
Rem PRVLG
    If @@34@ > @@35@ then
      Gyou = Soto_Cnt
      Retu = Uti_Cnt
    End If
    Do Until Uti_Cnt = Uti_Max + 1
BOX_1:
Rem PRVLG
  If W@@30@_@@31@_@@28@_@@29@_@@02@ (Gyou, Retu) = CNS_NOT_KUH_@@23@ And _
Rem PRVLG
  W3_@@31@_@@28@_@@29@_@@02@ (Gyou, Retu) = @@07@ Then
    GoTo BOX_2
  End If
    GoTo BOX_E
BOX_2:
Rem PUBIN
  If W@@305@_@@315@_@@285@_@@295@_@@200@ (Gyou, Retu) = CNS_NOT_KUH_@@235@ Then Go To Box_3
Rem PRVLG
  @@17@
BOX_3:
Rem PRVLG
  If W@@30@_@@31@_@@28@_@@29@_@@02@_wk (Gyou, Retu) ◇ CNS_NOT_KUH_@@23@ Then
    GoTo BOX_4
  End If
    GoTo BOX_5
BOX_4:
Rem PRVLG
  W@@30@_@@31@_@@28@_@@29@_@@02@ (Gyou, Retu) = W@@30@_@@31@_@@28@_@@29@_@@02@_wk (Gyou, Retu)
Rem PRVLG

```

## 【図 142】

付録一全ベクトルの型: 018-3

正規/出力/配列有/等価有/境界有のL4 (管理番号0180-3)

```
W0%300@_0%310@_0%280@_0%290@_0%200@ (Gyou, Retu) = W0%300@_0%310@_0%280@_0%290@_0%020@ (Gyou, Retu)
Rem PRVLG
  CTRL_W0%300@_0%280@_0%290@_STS_TRANSITION_FLG = CTRL_W0%300@_0%280@_0%290@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_W0%300@_0%280@_0%290@_STS_TRANSITION_FLG_P
    = CTRL_W0%300@_0%280@_0%290@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W0%300@_0%310@_0%280@_0%290@_0%020@_NOT_VALID_FLG (Gyou, Retu) = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W0%300@_0%290@_ING_FLG = True
  GoTo BOX_E
BOX_E:
  Uti_Cnt = Uti_Cnt + 1
Rem PRVLG
  If 0%340 < 0%350 then
    Gyou = Uti_Cnt
    Retu = Soto_Cnt
  End If
Rem PRVLG
  If 0%340 > 0%350 then
    Gyou = Soto_Cnt
    Retu = Uti_Cnt
  End If
  Loop
  Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
  If 0%340 < 0%350 then
    Gyou = Uti_Cnt
    Retu = Soto_Cnt
  End If
Rem PRVLG
  If 0%340 > 0%350 then
    Gyou = Soto_Cnt
    Retu = Uti_Cnt
  End If
  Loop
  Gyou = 0
  Retu = 0
  Uti_Cnt = 0
  Soto_Cnt = 0
  Uti_Max = 0
  Soto_Max = 0
End Sub
Rem *** PRCEND
```

## 【図 143】

付録一全ベクトルの型: 019-1

K/出力/配列無/等価無/境界無のL3 (管理番号0190-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
Private W0%300_0%310_0%280_0%290_0%020 As Integer
Rem PUB1N3
Private W0%3050_0%3150_0%2850_0%2950_0%2000 As 0%2350
Rem PUB5C3
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRI
Rem PRV2T3
Private W0%300_0%310_0%280_0%290_0%020_wk As Integer
Rem PRV7R3
Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_Integer As Integer
Rem $PRV2W3
0%1620
Rem $PRV2E3
Rem $PRV4W3
0%1650
Rem $PRV4E3
Rem #DEFEND
Public Sub Main 0
Rem #VECREP Private Sub L3_0%310_0%280_0%290_0%020 0
BOX_1:
Rem PRVLG
If W0%300_0%310_0%280_0%290_0%020 = CNS_NOT_KUH_Integer Then
GoTo BOX_2
End If
GoTo BOX_E
BOX_2:
Rem PRVLG
0%160
BOX_3:
Rem PRVLG
If W0%300_0%310_0%280_0%290_0%020_wk <> CNS_NOT_KUH_Integer Then
GoTo BOX_4
End If
GoTo BOX_5
BOX_4:
Rem PRVLG
W0%300_0%310_0%280_0%290_0%020 = W0%300_0%310_0%280_0%290_0%020_wk
Rem PRVLG
CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG = CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG + 1
GoTo BOX_E
BOX_5:
```

## 【図 1 4 4】

付録一全ベクトルの型: 019-2

K/出力/配列無/等価無/境界無のL3 (管理番号0190-2)

```
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 145】

付録-全ベクトルの型: 020-1

K/出力/配列無/等価無/境界無のL4 (管理番号0200-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W0%300_0%310_0%280_0%290_0%020_Sum As 0%230
Rem PUBIN4
Private W0%3050_0%3150_0%2850_0%2950_0%2000 As 0%2350
Rem PUB5C4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
Private CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem PUB4B4
' Private W0%3000_0%3100_0%2800_0%2900_0%2000 As 0%230
Rem PUB4N3
Private W3_0%310_0%280_0%290_0%020 As Integer
Rem PUB4N2
' Private W2_0%310_0%280_0%290_0%020 As 0%230
Rem #DEFPRV
Rem PRV2T4
Private W0%300_0%310_0%280_0%290_0%020_wk As 0%230
Rem PRV7R4
Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem *PUBKU
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant1 As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Rem $PRV2W4
0%1720
Rem $PRV2E4
Rem $PRV4W4
0%1820
Rem $PRV4E4
Rem #DEFEND
Public Sub Main 0
Rem #VECREP Private Sub L4_0%310_0%280_0%290_0%020 0
BOX_1:
Rem PRVLG
If W0%300_0%310_0%280_0%290_0%020_Sum = CNS_NOT_KUH_0%230 And _
Rem PRVLG
W3_0%310_0%280_0%290_0%020 = 0%070 Then
GoTo BOX_2
End If
```



## 【図 1 4 6】

付録-全ベクトルの型: 0 2 0 - 2

K/出力/配列無/等価無/境界無の L 4 (管理番号 0 2 0 0 - 2)

```
GoTo BOX_E
BOX_2:
Rem PUBIN
  If W@%305@_@%315@_@%285@_@%295@_@%200@ = CNS_NOT_KUH_@%235@ Then Go To Box_3
Rem PRVLG
  @%17@
BOX_3:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@_wk <> CNS_NOT_KUH_@%23@ Then
    GoTo BOX_4
  End If
  GoTo BOX_5
BOX_4:
Rem PRVLG
  W@%30@_@%31@_@%28@_@%29@_@%02@_Sum = W@%30@_@%31@_@%28@_@%29@_@%02@_wk
Rem PRVLG
  W2_@%31@_@%28@_@%29@_@%02@ = W@%30@_@%31@_@%28@_@%29@_@%02@_wk
Rem PRVLG
  CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 147】

付録-全ベクトルの型: 021-1

K/出力/配列無/等価有/境界無のL3 (管理番号0210-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
  Private W@%30@_@%31@_@%28@_@%29@_@%02@ As Integer
Rem PUBIN3
  Private W@%305@_@%315@_@%285@_@%295@_@%200@ As @%235@
Rem PUB5C3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
  Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
  Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ As Integer
Rem PRV7R3
  Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_Integer As Integer
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
Public Sub Main()
Rem #VECREP Private Sub L3_@%31@_@%28@_@%29@_@%02@()
  BOX_1:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@ = CNS_NOT_KUH_Integer Then
    GoTo BOX_2
  End If
  GoTo BOX_E
  BOX_2:
Rem PRVLG
  @%16@
  BOX_3:
Rem PUBL3F
  If W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ = 1 Then GoTo BOX_4
  GoTo BOX_5
  BOX_4:
Rem PUBL3F
  If W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ = 1 Then W@%30@_@%31@_@%28@_@%29@_@%02@ = @%07@
Rem PRVLG
  CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
  BOX_5:
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
```

## 【図 148】

付録—全ベクトルの型: 021-2

K/出力/配列無/等価有/境界無のL3 (管理番号0210-2)

```
= CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG_PP Then
  GoTo BOX_6
End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_%31@_%28@_%29@_%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 149】

付録—全ベクトルの型: 022-1

K/出力/配列無/等価有/境界無のL4 (管理番号0220-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
  Private W0%300_0%310_0%280_0%290_0%020_Sum As 0%230
Rem PUBIN4
  Private W0%3050_0%3150_0%2850_0%2950_0%2000 As 0%2350
Rem PUB5C4
  Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
  Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
  Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
  Private CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem PUB4B4
  ' Private W0%3000_0%3100_0%2800_0%2900_0%2000 As 0%230
Rem PUB4N3
  Private W3_0%310_0%280_0%290_0%020 As Integer
Rem PUB4N2
  ' Private W2_0%310_0%280_0%290_0%020 As 0%230
Rem #DEFPRV
Rem PRV2T4
  Private W0%300_0%310_0%280_0%290_0%020_wk As 0%230
Rem PRV7R4
  Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem *PUBKU
  Private CNS_NOT_KUH_String As String
  Private CNS_NOT_KUH_Integer As Integer
  Private CNS_NOT_KUH_Boolean As Boolean
  Private CNS_NOT_KUH_Long As Long
  Private CNS_NOT_KUH_Single As Single
  Private CNS_NOT_KUH_Double As Double
  Private CNS_NOT_KUH_Variant As Variant
  Private CNS_NOT_KUH_Currency As Currency
  Private CNS_NOT_KUH_Byte As Byte
  Private CNS_NOT_KUH_Date As Date
Rem $PRV2W4
0%1720
Rem $PRV2E4
Rem $PRV4W4
0%1820
Rem $PRV4E4
Rem #DEFEND
  Public Sub Main()
Rem #VECREP Private Sub L4_0%310_0%280_0%290_0%020_0%070 0
  BOX_1:
Rem PRVLG
    If W0%300_0%310_0%280_0%290_0%020_Sum = CNS_NOT_KUH_0%230 And _
Rem PRVLG
    W3_0%310_0%280_0%290_0%020 = 0%070 Then
      GoTo BOX_2
    End If
```

## 【図 150】

付録-全ベクトルの型: 022-2

K/出力/配列無/等価有/境界無のL4 (管理番号0220-2)

```
GoTo BOX_E
BOX_2:
Rem PUBIN
  If W@%305@_%315@_%285@_%295@_%200@ = CNS_NOT_KUH_%235@ Then Go To Box_3
Rem PRVLG
  @%17@
BOX_3:
Rem PRVLG
  If W@%30@_%31@_%28@_%29@_%02@_wk <> CNS_NOT_KUH_%23@ Then
    GoTo BOX_4
  End If
  GoTo BOX_5
BOX_4:
Rem PRVLG
  W@%30@_%31@_%28@_%29@_%02@_Sum = W@%30@_%31@_%28@_%29@_%02@_wk
Rem PRVLG
  W2_@%31@_%28@_%29@_%02@ = W@%30@_%31@_%28@_%29@_%02@_wk
Rem PRVLG
  CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG = CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_%31@_%28@_%29@_%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 1 5 1】

付録-全ベクトルの型: 0 2 3 - 1

K/出力/配列有/等価無/境界無のL 3 (管理番号 0 2 3 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
  Private W@%30@_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As Integer
Rem PUBIN3
  Private W@%305@_@%315@_@%285@_@%295@_@%200@ (@%265@, @%275@) As @%235@
Rem PUB5C3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
  Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (@%26@, @%27@) As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
  Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk (@%26@, @%27@) As Integer
Rem PRV7R3
  Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_Integer As Integer
  Private Gyou as Integer
  Private Retu as Integer
  Private Uti_Cnt as Integer
  Private Soto_Cnt as Integer
  Private Uti_Max as Integer
  Private Soto_Max as Integer
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
  Public Sub Main ()
Rem #VECREP Private Sub L3_@%31@_@%28@_@%29@_@%02@ ()
  Gyou = 0
  Retu = 0
Rem PRVLG
  If @%34@ < @%35@ then
Rem PRVLG
    Uti_Max = @%26@
Rem PRVLG
    Soto_Max = @%27@
  End If
Rem PRVLG
  If @%34@ > @%35@ then
Rem PRVLG
    Uti_Max = @%27@
Rem PRVLG
    Soto_Max = @%26@
  End If
```

## 【図 1 5 2】

付録-全ベクトルの型: 0 2 3 - 2

K/出力/配列有/等価無/境界無のL 3 (管理番号 0 2 3 0 - 2)

```
      Uti_Cnt = 0
      Soto_Cnt = 0
      Do Until Soto_Cnt = Soto_Max + 1
        Uti_Cnt = 0
      Rem PRVLG
        If @%34@ < @%35@ then
          Gyou = Uti_Cnt
          Retu = Soto_Cnt
        End If
      Rem PRVLG
        If @%34@ > @%35@ then
          Gyou = Soto_Cnt
          Retu = Uti_Cnt
        End If
        Do Until Uti_Cnt = Uti_Max + 1
      BOX_1:
      Rem PRVLG
        If W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = CNS_NOT_KUH_Integer Then
          GoTo BOX_2
        End If
        GoTo BOX_E
      BOX_2:
      Rem PRVLG
        @%16@
      BOX_3:
      Rem PRVLG
        If W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu) <> CNS_NOT_KUH_@%23@ Then
          GoTo BOX_4
        End If
        GoTo BOX_5
      BOX_4:
      Rem PRVLG
        W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu)
      Rem PRVLG
        CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
        GoTo BOX_E
      BOX_5:
      Rem PRVLG
        If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
          = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
          GoTo BOX_6
        End If
        GoTo BOX_7
      BOX_6:
      Rem PRVLG
        CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (Gyou, Retu) = True
        GoTo BOX_E
      BOX_7:
      Rem PRVLG
        CTRL_W@%30@_@%29@_ING_FLG = True
        GoTo BOX_E
      BOX_E:
        Uti_Cnt = Uti_Cnt + 1
```

## 【図 1 5 3】

付録-全ベクトルの型: 0 2 3 - 3

K/出力/配列有/等価無/境界無のL 3 (管理番号 0 2 3 0 - 2)

```
Rem PRVLG
    If @%34@ < @%35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If @%34@ > @%35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
    Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
    If @%34@ < @%35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If @%34@ > @%35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
    Gyou = 0
    Retu = 0
    Uti_Cnt = 0
    Soto_Cnt = 0
    Uti_Max = 0
    Soto_Max = 0
End Sub
Rem *** PRCEND
```



## 【図 1 5 4】

付録—全ベクトルの型: 0 2 4 - 1

K/出力/配列有/等価無/境界無のL 4 (管理番号 0 2 4 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W@%30@_@%31@_@%28@_@%29@_@%02@_Sum (@%26@, @%27@) As @%23@
Rem PUBIN4
Private W@%305@_@%315@_@%285@_@%295@_@%200@ (@%265@, @%275@) As @%235@
Rem PUB5C4
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (@%26@, @%27@) As Boolean
Rem #DEFEND
Rem PUB4B4
' Private W@%300@_@%310@_@%280@_@%290@_@%200@ (@%26@, @%27@) As @%23@
Rem PUB4N3
Private W3_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As Integer
Rem PUB4N2
' Private W2_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As @%23@
Rem #DEFPRV
Rem PRV2T4
Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk (@%26@, @%27@) As @%23@
Rem PRV7R4
Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem *PUBKU
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Private Gyou as Integer
Private Retu as Integer
Private Uti_Cnt as Integer
Private Soto_Cnt as Integer
Private Uti_Max as Integer
Private Soto_Max as Integer
Rem $PRV2W4
@%172@
Rem $PRV2E4
Rem $PRV4W4
@%182@
Rem $PRV4E4
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub L4_@%31@_@%28@_@%29@_@%02@0
Gyou = 0
```

## 【図 155】

付録一全ベクトルの型: 024-2

K/出力/配列有/等価無/境界無のL4 (管理番号0240-2)

```

    Retu = 0
Rem PRVLG
    If @%34@ < @%35@ then
Rem PRVLG
        Uti_Max = @%26@
Rem PRVLG
        Soto_Max = @%27@
    End If
Rem PRVLG
    If @%34@ > @%35@ then
Rem PRVLG
        Uti_Max = @%27@
Rem PRVLG
        Soto_Max = @%26@
    End If
    Uti_Cnt = 0
    Soto_Cnt = 0
    Do Until Soto_Cnt = Soto_Max + 1
        Uti_Cnt = 0
Rem PRVLG
        If @%34@ < @%35@ then
            Gyou = Uti_Cnt
            Retu = Soto_Cnt
        End If
Rem PRVLG
        If @%34@ > @%35@ then
            Gyou = Soto_Cnt
            Retu = Uti_Cnt
        End If
        Do Until Uti_Cnt = Uti_Max + 1
BOX_1:
Rem PRVLG
        If W@%30@_@%31@_@%28@_@%29@_@%02@_Sum (Gyou, Retu) = CNS_NOT_KUH_@%23@ And _
Rem PRVLG
            W3_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = @%07@ Then
                GoTo BOX_2
            End If
            GoTo BOX_E
BOX_2:
Rem PUBIN
        If W@%305@_@%315@_@%285@_@%295@_@%200@ (Gyou, Retu) = CNS_NOT_KUH_@%235@ Then Go To Box_3
Rem PRVLG
        @%17@
BOX_3:
Rem PRVLG
        If W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu) < CNS_NOT_KUH_@%23@ Then
            GoTo BOX_4
        End If
        GoTo BOX_5
BOX_4:
Rem PRVLG
        W@%30@_@%31@_@%28@_@%29@_@%02@_Sum (Gyou, Retu) = W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu)
Rem PRVLG

```

## 【図 156】

付録-全ベクトルの型: 024-3

K/出力/配列有/等価無/境界無のL4 (管理番号0240-3)

```
W2_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu)
Rem PRVLG
  CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (Gyou, Retu) = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
  Uti_Cnt = Uti_Cnt + 1
Rem PRVLG
  If @%34@ < @%35@ then
    Gyou = Uti_Cnt
    Retu = Soto_Cnt
  End If
Rem PRVLG
  If @%34@ > @%35@ then
    Gyou = Soto_Cnt
    Retu = Uti_Cnt
  End If
  Loop
  Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
  If @%34@ < @%35@ then
    Gyou = Uti_Cnt
    Retu = Soto_Cnt
  End If
Rem PRVLG
  If @%34@ > @%35@ then
    Gyou = Soto_Cnt
    Retu = Uti_Cnt
  End If
  Loop
  Gyou = 0
  Retu = 0
  Uti_Cnt = 0
  Soto_Cnt = 0
  Uti_Max = 0
  Soto_Max = 0
End Sub
Rem *** PRCEND
```

## 【図 1 5 7】

付録—全ベクトルの型: 0 2 5 - 1

K/出力/配列有/等価有/境界無のL 3 (管理番号 0 2 5 0 - 1)

```

Option Explicit
Rem #DEFPUB
Rem PUB4N3
Private W@%30@_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As Integer
Rem PUBIN3
Private W@%305@_@%315@_@%285@_@%295@_@%200@ (@%265@, @%275@) As @%235@
Rem PUB5C3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (@%26@, @%27@) As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ (@%26@, @%27@) As Integer
Rem PRV7R3
Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_Integer As Integer
Private Gyou as Integer
Private Retu as Integer
Private Uti_Cnt as Integer
Private Soto_Cnt as Integer
Private Uti_Max as Integer
Private Soto_Max as Integer
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
Public Sub Main ()
Rem #VECREP Private Sub L3_@%31@_@%28@_@%29@_@%02@ ()
    Gyou = 0
    Retu = 0
Rem PRVLG
    If @%34@ < @%35@ then
Rem PRVLG
        Uti_Max = @%26@
Rem PRVLG
        Soto_Max = @%27@
    End If
Rem PRVLG
    If @%34@ > @%35@ then
Rem PRVLG
        Uti_Max = @%27@
Rem PRVLG
        Soto_Max = @%26@
    End If

```

## 【図 158】

付録一全ベクトルの型: 025-2

K/出力/配列有/等価有/境界無のL3 (管理番号0250-2)

```

    Uti_Cnt = 0
    Soto_Cnt = 0
    Do Until Soto_Cnt = Soto_Max + 1
        Uti_Cnt = 0
    Rem PRVLG
        If 0x340 < 0x350 then
            Gyou = Uti_Cnt
            Retu = Soto_Cnt
        End If
    Rem PRVLG
        If 0x340 > 0x350 then
            Gyou = Soto_Cnt
            Retu = Uti_Cnt
        End If
        Do Until Uti_Cnt = Uti_Max + 1
    BOX_1:
    Rem PRVLG
        If W0x300_0x310_0x280_0x290_0x020 (Gyou, Retu) = CNS_NOT_KUH_Integer Then
            GoTo BOX_2
        End If
        GoTo BOX_E
    BOX_2:
    Rem PRVLG
        0x160
    BOX_3:
    Rem PUBL3F
        If W0x300_0x310_0x280_0x290_0x020_wk0x070 (Gyou, Retu) = 1 Then GoTo BOX_4
        GoTo BOX_5
    BOX_4:
    Rem PUBL3F
        If W0x300_0x310_0x280_0x290_0x020_wk0x070 (Gyou, Retu) = 1 Then W0x300_0x310_0x280_0x290_0x020 (Gyou, Retu) = 6x0'
    Rem PRVLG
        CTRL_W0x300_0x280_0x290_STS_TRANSITION_FLG = CTRL_W0x300_0x280_0x290_STS_TRANSITION_FLG + 1
        GoTo BOX_E
    BOX_5:
    Rem PRVLG
        If CTRL_W0x300_0x280_0x290_STS_TRANSITION_FLG_P
            = CTRL_W0x300_0x280_0x290_STS_TRANSITION_FLG_PP Then
            GoTo BOX_6
        End If
        GoTo BOX_7
    BOX_6:
    Rem PRVLG
        CTRL_W0x300_0x310_0x280_0x290_0x020_NOT_VALID_FLG (Gyou, Retu) = True
        GoTo BOX_E
    BOX_7:
    Rem PRVLG
        CTRL_W0x300_0x290_ING_FLG = True
        GoTo BOX_E
    BOX_E:
        Uti_Cnt = Uti_Cnt + 1
    Rem PRVLG
        If 0x340 < 0x350 then

```

## 【図 1 5 9】

付録一全ベクトルの型: 0 2 5 - 3

K / 出力 / 配列有 / 等価有 / 境界無の L 3 (管理番号 0 2 5 0 - 3)

```
Gyou = Uti_Cnt
Retu = Soto_Cnt
End If
Rem PRVLG
If @@34@ > @@35@ then
    Gyou = Soto_Cnt
    Retu = Uti_Cnt
End If
Loop
Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
If @@34@ < @@35@ then
    Gyou = Uti_Cnt
    Retu = Soto_Cnt
End If
Rem PRVLG
If @@34@ > @@35@ then
    Gyou = Soto_Cnt
    Retu = Uti_Cnt
End If
Loop
Gyou = 0
Retu = 0
Uti_Cnt = 0
Soto_Cnt = 0
Uti_Max = 0
Soto_Max = 0
End Sub
Rem *** PRCEND
```

## 【図 160】

付録—全ベクトルの型: 026-1

K/出力/配列有/等価有/境界無のL4 (管理番号0260-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W0%300_@%310_@%280_@%290_@%020_Sum (@%260, @%270) As @%230
Rem PUBIN4
Private W0%3050_@%3150_@%2850_@%2950_@%2000 (@%2650, @%2750) As @%2350
Rem PUB5C4
Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
Private CTRL_W0%300_@%310_@%280_@%290_@%020_NOT_VALID_FLG (@%260, @%270) As Boolean
Rem #DEFEND
Rem PUB4B4
Private W0%3000_@%3100_@%2800_@%2900_@%2000 (@%260, @%270) As @%230
Rem PUB4N3
Private W3_@%310_@%280_@%290_@%020 (@%260, @%270) As Integer
Rem PUB4N2
Private W2_@%310_@%280_@%290_@%020 (@%260, @%270) As @%230
Private Gyou as Integer
Private Retu as Integer
Private Uti_Cnt as Integer
Private Soto_Cnt as Integer
Private Uti_Max as Integer
Private Soto_Max as Integer
Rem #DEFPRV
Rem PRV2T4
Private W0%300_@%310_@%280_@%290_@%020_wk (@%260, @%270) As @%230
Rem PRV7R4
Private CTRL_W0%300_@%290_ING_FLG As Boolean
Rem *PUBKU
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Rem $PRV2W4
@%1720
Rem $PRV2E4
Rem $PRV4W4
@%1820
Rem $PRV4E4
Rem #DEFEND
Public Sub Main 0
Rem #VECREP Private Sub L4_@%310_@%280_@%290_@%020_@%0700
Gyou = 0
```

## 【図 1 6 1】

付録—全ベクトルの型 : 0 2 6 - 2

K / 出力 / 配列有 / 等価有 / 境界無の L 4 (管理番号 0 2 6 0 - 2)

```

    Retu = 0
Rem PRVLG
    If @%34@ < @%35@ then
Rem PRVLG
        Uti_Max = @%26@
Rem PRVLG
        Soto_Max = @%27@
    End If
Rem PRVLG
    If @%34@ > @%35@ then
Rem PRVLG
        Uti_Max = @%27@
Rem PRVLG
        Soto_Max = @%26@
    End If
    Uti_Cnt = 0
    Soto_Cnt = 0
    Do Until Soto_Cnt = Soto_Max + 1
        Uti_Cnt = 0
Rem PRVLG
        If @%34@ < @%35@ then
            Gyou = Uti_Cnt
            Retu = Soto_Cnt
        End If
Rem PRVLG
        If @%34@ > @%35@ then
            Gyou = Soto_Cnt
            Retu = Uti_Cnt
        End If
        Do Until Uti_Cnt = Uti_Max + 1
BOX_1:
Rem PRVLG
        If W@%30@_@%31@_@%28@_@%29@_@%02@_Sum (Gyou, Retu) = CNS_NOT_KUH_@%23@ And _
Rem PRVLG
            W3_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = @%07@ Then
                GoTo BOX_2
            End If
            GoTo BOX_E
BOX_2:
Rem PUBIN
            If W@%305@_@%315@_@%285@_@%295@_@%200@ (Gyou, Retu) = CNS_NOT_KUH_@%235@ Then Go To Box_3
Rem PRVLG
            @%17@
BOX_3:
Rem PRVLG
            If W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu) <> CNS_NOT_KUH_@%23@ Then
                GoTo BOX_4
            End If
            GoTo BOX_5
BOX_4:
Rem PRVLG
            W@%30@_@%31@_@%28@_@%29@_@%02@_Sum (Gyou, Retu) = W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu)
Rem PRVLG

```



## 【図 162】

付録—全ベクトルの型: 026-3

K/出力/配列有/等価有/境界無のL4 (管理番号0260-3)

```
W2_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu)
Rem PRVLG
    CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
    GoTo BOX_E
BOX_5:
Rem PRVLG
    If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
        = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
        GoTo BOX_6
    End If
    GoTo BOX_7
BOX_6:
Rem PRVLG
    CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (Gyou, Retu) = True
    GoTo BOX_E
BOX_7:
Rem PRVLG
    CTRL_W@%30@_@%29@_ING_FLG = True
    GoTo BOX_E
BOX_E:
    Uti_Cnt = Uti_Cnt + 1
Rem PRVLG
    If @%34@ < @%35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If @%34@ > @%35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
    Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
    If @%34@ < @%35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If @%34@ > @%35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
Gyou = 0
Retu = 0
Uti_Cnt = 0
Soto_Cnt = 0
Uti_Max = 0
Soto_Max = 0
End Sub
Rem *** PRCEND
```

## 【図 163】

付録—全ベクトルの型: 027-1

M/出力/配列無/等価無/境界無のL3 (管理番号0290-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
Private W%300_@%310_@%280_@%290_@%020 As Integer
Rem PUBIN3
Private W%3050_@%3150_@%2850_@%2950_@%2000 As @%2350
Rem PUB5C3
Private CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_W%300_@%310_@%280_@%290_@%020_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
Private W%300_@%310_@%280_@%290_@%020_wk As Integer
Rem PRV7R3
Private CTRL_W%300_@%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_Integer As Integer
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
Public Sub Main()
Rem #VECREP Private Sub L3_@%310_@%280_@%290_@%020()
BOX_1:
Rem PRVLG
If W%300_@%310_@%280_@%290_@%020 = CNS_NOT_KUH_Integer Then
GoTo BOX_2
End If
GoTo BOX_E
BOX_2:
Rem PRVLG
@%16@
BOX_3:
Rem PRVLG
If W%300_@%310_@%280_@%290_@%020_wk < CNS_NOT_KUH_Integer Then
GoTo BOX_4
End If
GoTo BOX_5
BOX_4:
Rem PRVLG
W%300_@%310_@%280_@%290_@%020 = W%300_@%310_@%280_@%290_@%020_wk
Rem PRVLG
CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG = CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG + 1
GoTo BOX_E
BOX_5:
```

## 【図 164】

付録—全ベクトルの型: 027-2

M/出力/配列無/等価無/境界無のL3 (管理番号0290-2)

```
Rem PRVLG
  If CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P
    = CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W0%300_0%290_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 165】

付録一全ベクトルの型: 028-1

M/出力/配列無/等価無/境界無のL4 (管理番号0300-1)

```

Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W0%300_@%310_@%280_@%290_@%020 As @%230
Rem PUBIN4
Private W0%3050_@%3150_@%2850_@%2950_@%2000 As @%2350
Rem PUB5C4
Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
Private CTRL_W0%300_@%310_@%280_@%290_@%020_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem PUB4B4
Private W0%3000_@%3100_@%2800_@%2900_@%2000 As @%230
Rem PUB4N3
Private W3_@%310_@%280_@%290_@%020 As Integer
Rem PUB4N2
Private W2_@%310_@%280_@%290_@%020 As @%230
Rem #DEFPRV
Rem PRV2T4
Private W0%300_@%310_@%280_@%290_@%020_wk As @%230
Rem PRV7R4
Private CTRL_W0%300_@%290_ING_FLG As Boolean
Rem *PUBKU
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Rem $PRV2W4
@%1720
Rem $PRV2E4
Rem $PRV4W4
@%1820
Rem $PRV4E4
Rem #DEFEND
Public Sub Main 0
Rem #VECREP Private Sub L4_@%310_@%280_@%290_@%020 0
BOX_1:
Rem PRVLG
If W0%300_@%310_@%280_@%290_@%020 = CNS_NOT_KUH_@%230
And W3_@%310_@%280_@%290_@%020 = @%070 Then
GoTo BOX_2
End If
GoTo BOX_E

```

## 【図 166】

付録一全ベクトルの型: 028-2

M/出力/配列無/等価無/境界無のL4 (管理番号0300-2)

```
BOX_2:
Rem PUBIN
  If W@%305@_@%315@_@%285@_@%295@_@%200@ = CNS_NOT_KUH_@%235@ Then Go To Box_3
Rem PRVLG
  @%17@
BOX_3:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@_wk <> CNS_NOT_KUH_@%23@ Then
    GoTo BOX_4
  End If
  GoTo BOX_5
BOX_4:
Rem PRVLG
  W@%30@_@%31@_@%28@_@%29@_@%02@ = W@%30@_@%31@_@%28@_@%29@_@%02@_wk
Rem PRVLG
  CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 1 6 7】

付録—全ベクトルの型: 0 2 9 - 1

M/出力/配列無/等価有/境界無の L 3 (管理番号 0 3 1 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
  Private W@%30@_@%31@_@%28@_@%29@_@%02@ As Integer
Rem PUB1N3
  Private W@%305@_@%315@_@%285@_@%295@_@%200@ As @%235@
Rem PUB5C3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
  Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
  Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ As Integer
Rem PRV7R3
  Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_Integer As Integer
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
  Public Sub Main 0
Rem #VECREP Private Sub L3_@%31@_@%28@_@%29@_@%02@ 0
  BOX_1:
Rem PRVLG
    If W@%30@_@%31@_@%28@_@%29@_@%02@ = CNS_NOT_KUH_Integer Then
      GoTo BOX_2
    End If
      GoTo BOX_E
  BOX_2:
Rem PRVLG
    @%16@
  BOX_3:
Rem PUBL3F
    If W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ = 1 Then GoTo BOX_4
      GoTo BOX_5
  BOX_4:
Rem PUBL3F
    If W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ = 1 Then W@%30@_@%31@_@%28@_@%29@_@%02@ = @%07@
Rem PRVLG
    CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
      GoTo BOX_E
  BOX_5:
Rem PRVLG
    If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
```

## 【図 1 6 8】

付録－全ベクトルの型： 0 2 9 - 2

M / 出力 / 配列無 / 等価有 / 境界無の L 3 (管理番号 0 3 1 0 - 2)

```
= CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
  GoTo BOX_6
End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 169】

付録—全ベクトルの型: 030-1

M/出力/配列無/等価有/境界無のL4 (管理番号0320-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
  Private W@%30@_@%31@_@%28@_@%29@_@%02@ As @%23@
Rem PUBIN4
  Private W@%305@_@%315@_@%285@_@%295@_@%200@ As @%235@
Rem PUB5C4
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
  Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem PUB4B4
  Private W@%300@_@%310@_@%280@_@%290@_@%200@ As @%23@
Rem PUB4N3
  Private W3_@%31@_@%28@_@%29@_@%02@ As Integer
Rem PUB4N2
  Private W2_@%31@_@%28@_@%29@_@%02@ As @%23@
Rem #DEFPRV
Rem PRV2T4
  Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk As @%23@
Rem PRV7R4
  Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem *PUBKU
  Private CNS_NOT_KUH_String As String
  Private CNS_NOT_KUH_Integer As Integer
  Private CNS_NOT_KUH_Boolean As Boolean
  Private CNS_NOT_KUH_Long As Long
  Private CNS_NOT_KUH_Single As Single
  Private CNS_NOT_KUH_Double As Double
  Private CNS_NOT_KUH_Variant As Variant
  Private CNS_NOT_KUH_Currency As Currency
  Private CNS_NOT_KUH_Byte As Byte
  Private CNS_NOT_KUH_Date As Date
Rem $PRV2W4
@%172@
Rem $PRV2E4
Rem $PRV4W4
@%182@
Rem $PRV4E4
Rem #DEFEND
  Public Sub Main()
Rem #VECREP Private Sub L4_@%31@_@%28@_@%29@_@%02@_@%07@()
  BOX_1:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@ = CNS_NOT_KUH_@%23@ And _
Rem PRVLG
  W3_@%31@_@%28@_@%29@_@%02@ = @%07@ Then
    GoTo BOX_2
  End If
```



## 【図 170】

付録一全ベクトルの型: 030-2

M/出力/配列無/等価有/境界無のL4 (管理番号0320-2)

```
GoTo BOX_E
BOX_2:
Rem PUBIN
  If W@%305@_@%315@_@%285@_@%295@_@%200@ = CNS_NOT_KUH_@%235@ Then Go To Box_3
Rem PRVLG
  @%17@
BOX_3:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@_wk <> CNS_NOT_KUH_@%23@ Then
    GoTo BOX_4
  End If
  GoTo BOX_5
BOX_4:
Rem PRVLG
  W@%30@_@%31@_@%28@_@%29@_@%02@ = W@%30@_@%31@_@%28@_@%29@_@%02@_wk
Rem PRVLG
  CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 171】

付録-全ヘクトルの型: U31-1

M/出力/配列有/等価無/境界無のL3 (管理番号0330-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
Private W@%30@_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As Integer
Rem PUBIN3
Private W@%305@_@%315@_@%285@_@%295@_@%200@ (@%265@, @%275@) As @%235@
Rem PUB5C3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (@%26@, @%27@) As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk (@%26@, @%27@) As Integer
Rem PRV7R3
Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_Integer As Integer
Private Gyou as Integer
Private Retu as Integer
Private Uti_Cnt as Integer
Private Soto_Cnt as Integer
Private Uti_Max as Integer
Private Soto_Max as Integer
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
Public Sub Main()
Rem #VECREP Private Sub L3_@%31@_@%28@_@%29@_@%02@()
    Gyou = 0
    Retu = 0
Rem PRVLG
    If @%34@ < @%35@ then
Rem PRVLG
        Uti_Max = @%26@
Rem PRVLG
        Soto_Max = @%27@
    End If
Rem PRVLG
    If @%34@ > @%35@ then
Rem PRVLG
        Uti_Max = @%27@
Rem PRVLG
        Soto_Max = @%26@
    End If
```

## 【図 1 7 2】

付録一全ベクトルの型 : 0 3 1 - 2

M/出力/配列有/等価無/境界無のL 3 (管理番号 0 3 3 0 - 2)

```

      Uti_Cnt = 0
      Soto_Cnt = 0
      Do Until Soto_Cnt = Soto_Max + 1
        Uti_Cnt = 0
      Rem PRVLG
        If @%34@ < @%35@ then
          Gyou = Uti_Cnt
          Retu = Soto_Cnt
        End If
      Rem PRVLG
        If @%34@ > @%35@ then
          Gyou = Soto_Cnt
          Retu = Uti_Cnt
        End If
        Do Until Uti_Cnt = Uti_Max + 1
      BOX_1:
      Rem PRVLG
        If W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = CNS_NOT_KUH_Integer Then
          GoTo BOX_2
        End If
        GoTo BOX_E
      BOX_2:
      Rem PRVLG
        @%16@
      Rem PRVLG
        W@%30@_@%31@_@%28@_@%29@_@%02@_wk = @%07@
      BOX_3:
      Rem PRVLG
        If W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu) ◇ CNS_NOT_KUH_Integer Then
          GoTo BOX_4
        End If
        GoTo BOX_5
      BOX_4:
      Rem PRVLG
        W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu)
      Rem PRVLG
        CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
        GoTo BOX_E
      BOX_5:
      Rem PRVLG
        If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
          = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
          GoTo BOX_6
        End If
        GoTo BOX_7
      BOX_6:
      Rem PRVLG
        CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (Gyou, Retu) = True
        GoTo BOX_E
      BOX_7:
      Rem PRVLG
        CTRL_W@%30@_@%29@_ING_FLG = True
        GoTo BOX_E

```

## 【図 173】

付録-全ベクトルの型: 031-3

M/出力/配列有/等価無/境界無のL3 (管理番号0330-3)

```
BOX_E:
    Uti_Cnt = Uti_Cnt + 1
Rem PRVLG
    If @@34@ < @@35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If @@34@ > @@35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
    Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
    If @@34@ < @@35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If @@34@ > @@35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
    Gyou = 0
    Retu = 0
    Uti_Cnt = 0
    Soto_Cnt = 0
    Uti_Max = 0
    Soto_Max = 0
End Sub
Rem *** PRCEND
```

## 【図 174】

付録一全ベクトルの型: 032-1

M/出力/配列有/等価無/境界無のL4 (管理番号0340-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W0%300_@%310_@%280_@%290_@%020 (@%260, @%270) As @%230
Rem PUBIN4
Private W0%3050_@%3150_@%2850_@%2950_@%2000 (@%2650, @%2750) As @%2350
Rem PUB5C4
Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
Private CTRL_W0%300_@%310_@%280_@%290_@%020_NOT_VALID_FLG (@%260, @%270) As Boolean
Rem #DEFEND

Rem PUB4B4
Private W0%3000_@%3100_@%2800_@%2900_@%2000 (@%260, @%270) As @%230
Rem PUB4N3
Private W3_@%310_@%280_@%290_@%020 (@%260, @%270) As Integer
Rem PUB4N2
' Private W2_@%310_@%280_@%290_@%020 (@%260, @%270) As @%230
Rem #DEFPRV
Rem PRV2T4
Private W0%300_@%310_@%280_@%290_@%020_wk (@%260, @%270) As @%230
Rem PRV7R4
Private CTRL_W0%300_@%290_ING_FLG As Boolean
Rem *PUBKU
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Private Gyou as Integer
Private Retu as Integer
Private Uti_Cnt as Integer
Private Soto_Cnt as Integer
Private Uti_Max as Integer
Private Soto_Max as Integer
Rem $PRV2W4
@%1720
Rem $PRV2E4
Rem $PRV4W4
@%1820
Rem $PRV4E4
Rem #DEFEND
Public Sub Main()
Rem #VECREP Private Sub L4_@%300_@%310_@%280_@%290_@%020 0
```

## 【図 1 7 5】

付録一全ベクトルの型: 0 3 2 - 2

M/出力/配列有/等価無/境界無のL 4 (管理番号 0 3 4 0 - 2)

```
Gyou = 0
Retu = 0
Rem PRVLG
  If @%34@ < @%35@ then
Rem PRVLG
  Uti_Max = @%26@
Rem PRVLG
  Soto_Max = @%27@
  End If
Rem PRVLG
  If @%34@ > @%35@ then
Rem PRVLG
  Uti_Max = @%27@
Rem PRVLG
  Soto_Max = @%26@
  End If
  Uti_Cnt = 0
  Soto_Cnt = 0
  Do Until Soto_Cnt = Soto_Max + 1
    Uti_Cnt = 0
Rem PRVLG
    If @%34@ < @%35@ then
      Gyou = Uti_Cnt
      Retu = Soto_Cnt
    End If
Rem PRVLG
    If @%34@ > @%35@ then
      Gyou = Soto_Cnt
      Retu = Uti_Cnt
    End If
    Do Until Uti_Cnt = Uti_Max + 1
BOX_1:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = CNS_NOT_KUH_@%23@ And _
Rem PRVLG
  W3_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = @%07@ Then
    GoTo BOX_2
  End If
  GoTo BOX_E
BOX_2:
Rem PUBIN
  If W@%305@_@%315@_@%285@_@%295@_@%200@ (Gyou, Retu) = CNS_NOT_KUH_@%235@ Then Go To Box_3
Rem PRVLG
  @%17@
BOX_3:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu) <> CNS_NOT_KUH_@%23@ Then
    GoTo BOX_4
  End If
  GoTo BOX_5
BOX_4:
Rem PRVLG
  W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = W@%30@_@%31@_@%28@_@%29@_@%02@_wk (Gyou, Retu)
```

## 【図 176】

付録一全ベクトルの型: 032-3

M/出力/配列有/等価無/境界無のL4 (管理番号0340-3)

```
Rem PRVLG
    CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
    GoTo BOX_E
BOX_5:
Rem PRVLG
    If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
        = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
        GoTo BOX_6
    End If
    GoTo BOX_7
BOX_6:
Rem PRVLG
    CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (Gyou, Retu) = True
    GoTo BOX_E
BOX_7:
Rem PRVLG
    CTRL_W@%30@_@%29@_ING_FLG = True
    GoTo BOX_E
BOX_E:
    Uti_Cnt = Uti_Cnt + 1
Rem PRVLG
    If @%34@ < @%35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If @%34@ > @%35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
    Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
    If @%34@ < @%35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
    End If
Rem PRVLG
    If @%34@ > @%35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
    End If
    Loop
    Gyou = 0
    Retu = 0
    Uti_Cnt = 0
    Soto_Cnt = 0
    Uti_Max = 0
    Soto_Max = 0
End Sub
Rem *** PRCEND
```

## 【図 177】

付録—全ベクトルの型: 033-1

M/出力/配列有/等価有/境界無のL3 (管理番号0350-1)

```

Option Explicit
Rem #DEFPUB
Rem PUB4N3
Private W%300_@%310_@%280_@%290_@%020 (@%260, @%270) As Integer
Rem PUBIN3
Private W%3050_@%3150_@%2850_@%2950_@%2000 (@%2650, @%2750) As %2350
Rem PUB5C3
Private CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_W%300_@%310_@%280_@%290_@%020_NOT_VALID_FLG (@%260, @%270) As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
Private W%300_@%310_@%280_@%290_@%020_wk%070 (@%260, @%270) As Integer
Rem PRV7R3
Private CTRL_W%300_@%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_Integer As Integer
Private Gyou as Integer
Private Retu as Integer
Private Uti_Cnt as Integer
Private Soto_Cnt as Integer
Private Uti_Max as Integer
Private Soto_Max as Integer
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
Public Sub Main()
Rem #VECREP Private Sub L3_@%310_@%280_@%290_@%020 ()
    Gyou = 0
    Retu = 0
Rem PRVLG
    If @%340 < @%350 then
Rem PRVLG
        Uti_Max = @%260
Rem PRVLG
        Soto_Max = @%270
    End If
Rem PRVLG
    If @%340 > @%350 then
Rem PRVLG
        Uti_Max = @%270
Rem PRVLG
        Soto_Max = @%260
    End If

```



## 【図 178】

付録—全ベクトルの型: 033-2

M/出力/配列有/等価有/境界無のL3 (管理番号0350-2)

```
    Uti_Cnt = 0
    Soto_Cnt = 0
    Do Until Soto_Cnt = Soto_Max + 1
        Uti_Cnt = 0
    Rem PRVLG
        If @%34@ < @%35@ then
            Gyou = Uti_Cnt
            Retu = Soto_Cnt
        End If
    Rem PRVLG
        If @%34@ > @%35@ then
            Gyou = Soto_Cnt
            Retu = Uti_Cnt
        End If
        Do Until Uti_Cnt = Uti_Max + 1
    BOX_1:
    Rem PRVLG
        If W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = CNS_NOT_KUH_Integer Then
            GoTo BOX_2
        End If
        GoTo BOX_E
    BOX_2:
    Rem PRVLG
        @%16@
    BOX_3:
    Rem PUBL3F
        If W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ (Gyou, Retu) = 1 Then GoTo BOX_4
        GoTo BOX_5
    BOX_4:
    Rem PUBL3F
        If W@%30@_@%31@_@%28@_@%29@_@%02@_wk@%07@ (Gyou, Retu) = 1
            Then W@%30@_@%31@_@%28@_@%29@_@%02@ (Gyou, Retu) = @%07@
    Rem PRVLG
        CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
        GoTo BOX_E
    BOX_5:
    Rem PRVLG
        If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
            = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
            GoTo BOX_6
        End If
        GoTo BOX_7
    BOX_6:
    Rem PRVLG
        CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (Gyou, Retu) = True
        GoTo BOX_E
    BOX_7:
    Rem PRVLG
        CTRL_W@%30@_@%29@_ING_FLG = True
        GoTo BOX_E
    BOX_E:
        Uti_Cnt = Uti_Cnt + 1
    Rem PRVLG
```

## 【図 179】

付録-全ベクトルの型: 033-3

M/出力/配列有/等価有/境界無のL3 (管理番号0350-3)

```
      If @@34@ < @@35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
      End If
Rem PRVLG
      If @@34@ > @@35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
      End If
      Loop
      Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
      If @@34@ < @@35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
      End If
Rem PRVLG
      If @@34@ > @@35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
      End If
      Loop
      Gyou = 0
      Retu = 0
      Uti_Cnt = 0
      Soto_Cnt = 0
      Uti_Max = 0
      Soto_Max = 0
End Sub
Rem *** PCEND
```

## 【図 1 8 0】

付録—全ベクトルの型: 0 3 4 - 1

M/出力/配列有/等価有/境界無の L 4 (管理番号 0 3 6 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
  Private W@%30@_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As @%23@
Rem PUBIN4
  Private W@%305@_@%315@_@%285@_@%295@_@%200@ (@%265@, @%275@) As @%235@
Rem PUB5C4
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
  Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (@%26@, @%27@) As Boolean
Rem #DEFEND
Rem PUB4B4
  Private W@%300@_@%310@_@%280@_@%290@_@%200@ (@%26@, @%27@) As @%23@
Rem PUB4N3
  Private W3_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As Integer
Rem PUB4N2
  Private W2_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As @%23@
Rem #DEFPRV
Rem PRV2T4
  Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk (@%26@, @%27@) As @%23@
Rem PRV7R4
  Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem *PUBKU
  Private CNS_NOT_KUH_String As String
  Private CNS_NOT_KUH_Integer As Integer
  Private CNS_NOT_KUH_Boolean As Boolean
  Private CNS_NOT_KUH_Long As Long
  Private CNS_NOT_KUH_Single As Single
  Private CNS_NOT_KUH_Double As Double
  Private CNS_NOT_KUH_Variant As Variant
  Private CNS_NOT_KUH_Currency As Currency
  Private CNS_NOT_KUH_Byte As Byte
  Private CNS_NOT_KUH_Date As Date
  Private Gyou as Integer
  Private Retu as Integer
  Private Uti_Cnt as Integer
  Private Soto_Cnt as Integer
  Private Uti_Max as Integer
  Private Soto_Max as Integer
Rem $PRV2W4
@%172@
Rem $PRV2E4
Rem $PRV4W4
@%182@
Rem $PRV4E4
Rem #DEFEND
  Public Sub Main 0
Rem #VECREP Private Sub L4_@%31@_@%28@_@%29@_@%02@_@%07@ 0
  Gyou = 0
```

## 【図 181】

付録-全ベクトルの型: 034-2

M/出力/配列有/等価有/境界無のL4 (管理番号0360-2)

```
Retu = 0
Rem PRVLG
  If @@34@ < @@35@ then
Rem PRVLG
  Uti_Max = @@26@
Rem PRVLG
  Soto_Max = @@27@
  End If
Rem PRVLG
  If @@34@ > @@35@ then
Rem PRVLG
  Uti_Max = @@27@
Rem PRVLG
  Soto_Max = @@26@
  End If
  Uti_Cnt = 0
  Soto_Cnt = 0
  Do Until Soto_Cnt = Soto_Max + 1
    Uti_Cnt = 0
Rem PRVLG
    If @@34@ < @@35@ then
      Gyou = Uti_Cnt
      Retu = Soto_Cnt
    End If
Rem PRVLG
    If @@34@ > @@35@ then
      Gyou = Soto_Cnt
      Retu = Uti_Cnt
    End If
    Do Until Uti_Cnt = Uti_Max + 1
BOX_1:
Rem PRVLG
  If W@@30@_@@31@_@@28@_@@29@_@@02@ (Gyou, Retu) = CNS_NOT_KUH_@@23@ And _
Rem PRVLG
  W3_@@31@_@@28@_@@29@_@@02@ (Gyou, Retu) = @@07@ Then
    GoTo BOX_2
  End If
  GoTo BOX_E
BOX_2:
Rem PUBIN
  If W@@305@_@@315@_@@285@_@@295@_@@200@ (Gyou, Retu) = CNS_NOT_KUH_@@235@ Then Go To Box_3
Rem PRVLG
  @@17@
BOX_3:
Rem PRVLG
  If W@@30@_@@31@_@@28@_@@29@_@@02@_wk (Gyou, Retu) <> CNS_NOT_KUH_@@23@ Then
    GoTo BOX_4
  End If
  GoTo BOX_5
BOX_4:
Rem PRVLG
  W@@30@_@@31@_@@28@_@@29@_@@02@ (Gyou, Retu) = W@@30@_@@31@_@@28@_@@29@_@@02@_wk (Gyou, Retu)
Rem PRVLG
```

## 【図 182】

付録-全ベクトルの型: 034-3

M/出力/配列有/等価有/境界無のL4 (管理番号0360-3)

```
CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
GoTo BOX_E

BOX_5:
Rem PRVLG
If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
= CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
GoTo BOX_6
End If
GoTo BOX_7

BOX_6:
Rem PRVLG
CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (Gyou, Retu) = True
GoTo BOX_E

BOX_7:
Rem PRVLG
CTRL_W@%30@_@%29@_ING_FLG = True
GoTo BOX_E

BOX_E:
Uti_Cnt = Uti_Cnt + 1
Rem PRVLG
If @%34@ < @%35@ then
Gyou = Uti_Cnt
Retu = Soto_Cnt
End If
Rem PRVLG
If @%34@ > @%35@ then
Gyou = Soto_Cnt
Retu = Uti_Cnt
End If
Loop
Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
If @%34@ < @%35@ then
Gyou = Uti_Cnt
Retu = Soto_Cnt
End If
Rem PRVLG
If @%34@ > @%35@ then
Gyou = Soto_Cnt
Retu = Uti_Cnt
End If
Loop
Gyou = 0
Retu = 0
Uti_Cnt = 0
Soto_Cnt = 0
Uti_Max = 0
Soto_Max = 0
End Sub
Rem *** PRCEND
```

## 【図 183】

付録一全ベクトルの型: 035-1

入アクセスキーのL3 (管理番号0370-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
Private W@%30@_@%31@_@%28@_@%29@_IN_ACCESS_KEY As Integer
Rem PUBIN3
Private W@%305@_@%315@_@%285@_@%295@_@%200@ As @%235@
Rem PUB5C3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_W@%30@_@%31@_@%28@_@%29@_IN_ACCESS_KEY_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
Private W@%30@_@%31@_@%28@_@%29@_IN_ACCESS_KEY_wk@%07@ As Integer
Rem PRV7R3
Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_Integer As Integer
Private Gyou as Integer
Private Retu as Integer
Private Uti_Cnt as Integer
Private Soto_Cnt as Integer
Private Uti_Max as Integer
Private Soto_Max as Integer
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
Public Sub Main()
Rem #VECREP Private Sub L3_@%31@_@%28@_@%29@_IN_ACCESS_KEY()
BOX_1:
Rem PRVLG
If W@%30@_@%31@_@%28@_@%29@_IN_ACCESS_KEY = W@%30@_@%31@_@%28@_@%29@_IN_ACCESS_KEY Then
GoTo BOX_2
End If
GoTo BOX_E
BOX_2:
Rem PRVLG
@%16@
BOX_3:
Rem PUBL3F
If W@%30@_@%31@_@%28@_@%29@_IN_ACCESS_KEY_wk@%07@ = 1 Then GoTo BOX_4
GoTo BOX_5
BOX_4:
Rem PUBL3F
If W@%30@_@%31@_@%28@_@%29@_IN_ACCESS_KEY_wk@%07@ = 1 T
```

## 【図 1 8 4】

付録一全ベクトルの型: 0 3 5 - 2

入力アクセスキーの L 3 (管理番号 0 3 7 0 - 2)

```
hen W@%30@_@%31@_@%28@_@%29@_IN_ACCESS_KEY = @%07@
Rem PRVLG
  CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_IN_ACCESS_KEY_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 1 8 5】

付録－全ベクトルの型: 0 3 6－1

入力処理条件キーのL 3 (管理番号 0 3 9 0－1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
  Private W@%30@_@%31@_@%28@_@%29@_IN_Conditions_KEY As Integer
Rem PUBIN3
  Private W@%305@_@%315@_@%285@_@%295@_@%200@ As @%235@
Rem PUB5C3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
  Private CTRL_W@%30@_@%31@_@%28@_@%29@_IN_Conditions_KEY_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
  Private W@%30@_@%31@_@%28@_@%29@_IN_Conditions_KEY_wk@%07@ As Integer
Rem PRV7R3
  Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_Integer As Integer
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
  Public Sub Main 0
Rem #VECREP Private Sub L3_@%31@_@%28@_@%29@_IN_Conditions_KEY 0
  BOX_1:
Rem PRVLG
    If W@%30@_@%31@_@%28@_@%29@_IN_Conditions_KEY = W@%30@_@%31@_@%28@_@%29@_IN_Conditions_KEY
    Then
      GoTo BOX_2
    End If
      GoTo BOX_E
  BOX_2:
Rem PRVLG
    @%16@
  BOX_3:
Rem PUBL3F
    If W@%30@_@%31@_@%28@_@%29@_IN_Conditions_KEY_wk@%07@ = 1 Then GoTo BOX_4
      GoTo BOX_5
  BOX_4:
Rem PUBL3F
    If W@%30@_@%31@_@%28@_@%29@_IN_Conditions_KEY_wk@%07@ = 1
    Then W@%30@_@%31@_@%28@_@%29@_IN_Conditions_KEY = @%07@
Rem PRVLG
    CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
      GoTo BOX_E
  BOX_5:
```



## 【図 186】

付録一全ベクトルの型: 036-2

入力処理条件キーのL3 (管理番号0390-2)

```
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_IN_Conditions_KEY_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 1 8 7】

付録一全ベクトルの型: 0 3 7 - 1

出力アクセスキーのL 3 (管理番号 0 4 1 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
Private W@%30@_@%31@_@%28@_@%29@_OUT_ACCESS_KEY As Integer
Rem PUBIN3
Private W@%305@_@%315@_@%285@_@%295@_@%200@ As @%235@
Rem PUB5C3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_W@%30@_@%31@_@%28@_@%29@_OUT_ACCESS_KEY_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
Private W@%30@_@%31@_@%28@_@%29@_OUT_ACCESS_KEY_wk@%07@ As Integer
Rem PRV7R3
Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_Integer As Integer
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
Public Sub Main()
Rem #VECREP Private Sub L3_@%31@_@%28@_@%29@_OUT_ACCESS_KEY()
BOX_1:
Rem PRVLG
If W@%30@_@%31@_@%28@_@%29@_OUT_ACCESS_KEY = W@%30@_@%31@_@%28@_@%29@_OUT_ACCESS_KEY Then
GoTo BOX_2
End If
GoTo BOX_E
BOX_2:
Rem PRVLG
@%16@
BOX_3:
Rem PUBL3F
If W@%30@_@%31@_@%28@_@%29@_OUT_ACCESS_KEY_wk@%07@ = 1 Then GoTo BOX_4
GoTo BOX_5
BOX_4:
Rem PUBL3F
If W@%30@_@%31@_@%28@_@%29@_OUT_ACCESS_KEY_wk@%07@ = 1
Then W@%30@_@%31@_@%28@_@%29@_OUT_ACCESS_KEY = @%07@
Rem PRVLG
CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
GoTo BOX_E
BOX_5:
Rem PRVLG
```

## 【図 188】

付録-全ベクトルの型: 037-2

出力アクセスキーのL3 (管理番号0410-2)

```
If CTRL_W@%300_@%280_@%290_STS_TRANSITION_FLG_P
  = CTRL_W@%300_@%280_@%290_STS_TRANSITION_FLG_PP Then
  GoTo BOX_6
End If
GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%300_@%310_@%280_@%290_OUT_ACCESS_KEY_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%300_@%290_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 189】

付録一全ベクトルの型: 038-1

出力処理条件キーのL3 (管理番号0430-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
Private W%300_@%310_@%280_@%290_OUT_Conditions_KEY As Integer
Rem PUBIN3
Private W%3050_@%3150_@%2850_@%2950_@%2000 As %2350
Rem PUB5C3
Private CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_W%300_@%310_@%280_@%290_OUT_Conditions_KEY_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
Private W%300_@%310_@%280_@%290_OUT_Conditions_KEY_wk%070 As Integer
Rem PRV7R3
Private CTRL_W%300_@%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_Integer As Integer
Rem $PRV2W3
@%1620
Rem $PRV2E3
Rem $PRV4W3
@%1650
Rem $PRV4E3
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub L3_@%310_@%280_@%290_OUT_Conditions_KEY 0
BOX_1:
Rem PRVLG
If W%300_@%310_@%280_@%290_OUT_Conditions_KEY = W%300_@%310_@%280_@%290_OUT_Conditions_KEY
Then
GoTo BOX_2
End If
GoTo BOX_E
BOX_2:
Rem PRVLG
@%160
BOX_3:
Rem PUBL3F
If W%300_@%310_@%280_@%290_OUT_Conditions_KEY_wk%070 = 1 Then GoTo BOX_4
GoTo BOX_5
BOX_4:
Rem PUBL3F
If W%300_@%310_@%280_@%290_OUT_Conditions_KEY_wk%070 = 1
Then W%300_@%310_@%280_@%290_OUT_Conditions_KEY = @%070
Rem PRVLG
CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG = CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG + 1
GoTo BOX_E
BOX_5:
```

## 【図 190】

付録一全ベクトルの型: 038-2

出力処理条件キーのL3 (管理番号0430-2)

```
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_OUT_Conditions_KEY_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 191】

付録一全ベクトルの型: 039-1

入力コマンドの I 2 (管理番号 0450-1)

---

```
Option Explicit
Rem #DEFPUB
Rem PUB4N2
  Private CTR_@%31@_@%28@_@%29@_I2_END As Boolean
Rem PUB4N2
  Private CTR_@%31@_@%28@_@%29@_I2_CNT As Integer
Rem PUB4N2
  Private CTR_@%31@_@%28@_@%29@_I2_ERR As Boolean
Rem PUB4N2
  Private CTR_@%31@_@%28@_@%29@_I2_EOF As Boolean
Rem PUBIN2
  Private W@%305@_@%315@_@%285@_@%295@_@%200@ As @%235@
Rem PUB5C2
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y2
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B2
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem I2AKYARA
Rem I2PKYARA
Rem PUB6F2
  Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV7R2
  Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem $PRV2W2
  @%152@
Rem $PRV2E2
Rem $PRV4W2
  @%155@
Rem $PRV4E2
Rem #DEFEND
  Public Sub Main()
Rem #VECREP Private Sub I2_@%31@_@%28@_@%29@_@%02@_@%07@()
  BOX_1:
Rem I2PKYKUU
  If @%50@ < @%07@ Then Exit Sub
Rem I2AKYKUU
  If @%50@ < @%07@ Then Exit Sub
  BOX_2:
Rem PRVLG
  @%15@
Rem PRVLG
  CTR_@%31@_@%28@_@%29@_I2_END = False
Rem PRVLG
  CTR_@%31@_@%28@_@%29@_I2_ERR = False
  BOX_3:
Rem PRVLG
  If CTR_@%31@_@%28@_@%29@_I2_END = False Then
    GoTo BOX_4
  End If
  GoTo BOX_5
```

---

## 【図 1 9 2】

付録—全ベクトルの型: 0 3 9 - 2

入力コマンドの I 2 (管理番号 0 4 5 0 - 2)

```
BOX_4:
Rem PRVLG
    @%153@
Rem PRVLG
    CTR_@%31@_@%28@_@%29@_I2_END = True
Rem PRVLG
    CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
    GoTo BOX_E
BOX_5:
Rem PRVLG
    If CTR_@%31@_@%28@_@%29@_I2_END = False Then
        GoTo BOX_6
    End If
    GoTo BOX_7
BOX_6:
Rem PRVLG
    CTR_@%31@_@%28@_@%29@_I2_ERR = True
Rem PRVLG
    CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
    GoTo BOX_E
BOX_7:
Rem PRVLG
    CTRL_W@%30@_@%29@_ING_FLG = True
    GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 193】

付録一全ベクトルの型: 040-1

出力コマンドの04 (管理番号0460-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private CTR_@%31@_@%28@_@%29@_04_END As Boolean
Rem PUB4N4
Private CTR_@%31@_@%28@_@%29@_04_CNT As Integer
Rem PUB4N4
Private CTR_@%31@_@%28@_@%29@_04_ERR As Boolean
Rem PUBIN4
Private W@%305@_@%315@_@%285@_@%295@_@%200@ As @%235@
Rem PUB5C4
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem 04AKYARA
Rem 04PKYARA
Rem PUB6F4
Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV7R4
Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem $PRV2W4
@%172@
Rem $PRV2E4
Rem $PRV4W4
@%182@
Rem $PRV4E4
Rem #DEFEND
Public Sub Main()
Rem #VECREP Private Sub 04_@%31@_@%28@_@%29@_@%02@_@%07@()
BOX_1:
Rem 04AKYKUU
If @%50@ < @%07@ Then Exit Sub
Rem 04PKYKUU
If @%50@ < @%07@ Then Exit Sub
BOX_2:
Rem PRVLG
@%17@
,
Rem PRVLG
CTR_@%31@_@%28@_@%29@_04_END = False
Rem PRVLG
CTR_@%31@_@%28@_@%29@_04_ERR = False
BOX_3:
Rem PRVLG
If CTR_@%31@_@%28@_@%29@_04_END = False Then
GoTo BOX_4
End If
GoTo BOX_5
BOX_4:
```



## 【図 1 9 4】

付録一全ベクトルの型： 0 4 0 - 2

出力コマンドの 0 4 (管理番号 0 4 6 0 - 2)

---

```
Rem PRVLG
    @%153@
Rem PRVLG
    CTR_@%31@_@%28@_@%29@_04_END = True
Rem PRVLG
    CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
    GoTo BOX_E
BOX_5:
Rem PRVLG
    If CTR_@%31@_@%28@_@%29@_04_END = False Then
        GoTo BOX_6
    End If
    GoTo BOX_7
BOX_6:
Rem PRVLG
    CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
Rem PRVLG
    CTR_@%31@_@%28@_@%29@_04_ERR = True
    GoTo BOX_E
BOX_7:
Rem PRVLG
    CTRL_W@%30@_@%29@_ING_FLG = True
    GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

---

## 【図 195】

付録一全ベクトルの型 : 041-1

経路の R4 (管理番号 0470-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB5C4
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem *PUBRT
Private CTR_NEXT_PALLET_ID_NEW As String
Rem PRV7R4
Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_String As String
Rem $PRV2W4
@%172@
Rem $PRV2E4
Rem $PRV4W4
@%182@
Rem $PRV4E4
Rem #DEFEND
Public Sub Main 0
Rem #VECREP Private Sub R4_@%31@_@%28@_@%29@_@%02@()
BOX_1:
Rem PRVLG
If CTR_NEXT_PALLET_ID_NEW = CNS_NOT_KUH_String Then
GoTo BOX_2
End If
GoTo BOX_E
BOX_2:
Rem PRVLG
CTR_NEXT_PALLET_ID_NEW = '@%100@'
BOX_3:
Rem PRVLG
If CTR_NEXT_PALLET_ID_NEW <> CNS_NOT_KUH_String Then
GoTo BOX_4
End If
GoTo BOX_5
BOX_4:
Rem PRVLG
CTR_NEXT_PALLET_ID_NEW = CTR_NEXT_PALLET_ID_NEW
GoTo BOX_E
BOX_5:
Rem PRVLG
If CTR_NEXT_PALLET_ID_NEW = CNS_NOT_KUH_String Then
GoTo BOX_6
End If
GoTo BOX_7
BOX_6:
```

【図 1 9 6】

付録一全ベクトルの型： 0 4 1 - 2

経路の R 4 (管理番号 0 4 7 0 - 2)

```
Rem PRVLG
    CTRL_W@%30@_%31@_%28@_%29@_%02@_NOT_VALID_FLG = True
    GoTo BOX_E
BOX_7:
Rem PRVLG
    CTRL_W@%30@_%29@_ING_FLG = True
    GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 197】

付録一全ベクトルの型: 042-1

経路の R2C (管理番号 0480-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB5C2
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y2
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B2
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F2
  Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem *PUBRT
  Private CTR_NEXT_PALLET_ID_NEW As String
Rem PRV7R2
  Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_String As String
Rem $PRV2W2
@%152@
Rem $PRV2E2
Rem $PRV4W2
@%155@
Rem $PRV4E2
Rem #DEFEND
  Public Sub Main()
Rem #VECREP Private Sub R2C_@%31@_@%28@_@%29@_@%02@()
  BOX_1:
Rem PRVLG
  If CTR_NEXT_PALLET_ID_NEW = CNS_NOT_KUH_String Then
    GoTo BOX_2
  End If
  GoTo BOX_E
  BOX_2:
Rem PRVLG
  @%15@
Rem PRVLG
  CTR_NEXT_PALLET_ID_NEW = '@%100@'
  BOX_3:
Rem PRVLG
  If CTR_NEXT_PALLET_ID_NEW <> CNS_NOT_KUH_String Then
    GoTo BOX_4
  End If
  GoTo BOX_5
  BOX_4:
Rem PRVLG
  @%153@
Rem PRVLG
  CTR_NEXT_PALLET_ID_NEW = CTR_NEXT_PALLET_ID_NEW
  GoTo BOX_E
  BOX_5:
Rem PRVLG
  If CTR_NEXT_PALLET_ID_NEW = CNS_NOT_KUH_String Then
```

## 【図 1 9 8】

付録一全ベクトルの型: 0 4 2 - 2

経路の R 2 C (管理番号 0 4 8 0 - 2)

```
      GoTo BOX_6
    End If
      GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_%31@_%28@_%29@_%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 199】

付録一全ベクトルの型: 043-1  
経路のR2 (管理番号0490-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB5C2
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y2
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B2
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F2
Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem #PUBRT
Private CTR_NEXT_PALLET_ID_NEW As String
Rem PRV7R2
Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_String As String
Rem $PRV2W2
@%152@
Rem $PRV2E2
Rem $PRV4W2
@%155@
Rem $PRV4E2
Rem #DEFEND
Public Sub Main()
Rem #VECREP Private Sub R2_@%31@_@%28@_@%29@_@%02@()
BOX_1:
Rem PRVLG
If CTR_NEXT_PALLET_ID_NEW = CNS_NOT_KUH_String Then
GoTo BOX_2
End If
GoTo BOX_E
BOX_2:
Rem PRVLG
CTR_NEXT_PALLET_ID_NEW = "@%100@"
BOX_3:
Rem PRVLG
If CTR_NEXT_PALLET_ID_NEW <> CNS_NOT_KUH_String Then
GoTo BOX_4
End If
GoTo BOX_5
BOX_4:
Rem PRVLG
CTR_NEXT_PALLET_ID_NEW = CTR_NEXT_PALLET_ID_NEW
GoTo BOX_E
BOX_5:
Rem PRVLG
If CTR_NEXT_PALLET_ID_NEW = CNS_NOT_KUH_String Then
GoTo BOX_6
End If
GoTo BOX_7
BOX_6:
```

## 【図 2 0 0】

付録－全ベクトルの型: 0 4 3 - 2

経路の R 2 (管理番号 0 4 9 0 - 2)

```
Rem PRVLG
    CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
    GoTo BOX_E
BOX_7:
Rem PRVLG
    CTRL_W@%30@_@%29@_ING_FLG = True
    GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 201】

付録一全ベクトルの型: 044-1

経路の R3R (管理番号 0500-1)

```

Option Explicit
Rem #DEFPUB
Rem PUB5C3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB5C2
Private CTRL_W2_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5C4
Private CTRL_W4_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB6F3
Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem #PUBRT
Private CTR_NEXT_PALLET_ID_NEW As String
Rem PRV7R3
Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_String As String
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
Public Sub Main()
Rem #VECREP Private Sub R3R_@%31@_@%28@_@%29@_@%02@()
BOX_1:
Rem PRVLG
If CTR_NEXT_PALLET_ID_NEW = CNS_NOT_KUH_String And _
Rem PRVLG
(CTRL_W4_@%28@_@%29@_STS_TRANSITION_FLG <> 0 Or _
Rem PRVLG
CTRL_W2_@%28@_@%29@_STS_TRANSITION_FLG <> 0) Then
GoTo BOX_2
End If
GoTo BOX_E
BOX_2:
Rem PRVLG
CTR_NEXT_PALLET_ID_NEW = '@%100@'
BOX_3:
Rem PRVLG
If CTR_NEXT_PALLET_ID_NEW <> CNS_NOT_KUH_String Then
GoTo BOX_4
End If
GoTo BOX_5
BOX_4:
Rem PRVLG
CTR_NEXT_PALLET_ID_NEW = CTR_NEXT_PALLET_ID_NEW

```



## 【図 2 0 2】

付録－全ベクトルの型： 0 4 4 - 2

経路の R 3 R (管理番号 0 5 0 0 - 2)

```
      GoTo BOX_E
BOX_5:
Rem PRVLG
      If CTR_NEXT_PALLET_ID_NEW = CNS_NOT_KUH_String Then
          GoTo BOX_6
      End If
      GoTo BOX_7
BOX_6:
Rem PRVLG
      CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
      GoTo BOX_E
BOX_7:
Rem PRVLG
      CTRL_W@%30@_@%29@_ING_FLG = True
      GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 2 0 3】

付録－全ベクトルの型： 0 4 5 - 1

正規（Kの派生元）／出力／配列無／等価無／境界無の L 3（管理番号 0 5 1 0 - 1）

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
  Private W@%30@_@%31@_@%28@_@%29@_@%02@ As Integer
Rem PUBIN3
  Private W@%305@_@%315@_@%285@_@%295@_@%200@ As @%235@
Rem PUB5C3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
  Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
  Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk As Integer
Rem PRV7R3
  Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_Integer As Integer
Rem $PRV2W3
@%162@
Rem $PRV2E3
Rem $PRV4W3
@%165@
Rem $PRV4E3
Rem #DEFEND
  Public Sub Main()
Rem #VECREP Private Sub L3_@%31@_@%28@_@%29@_@%02@()
  BOX_1:
Rem PRVLG
    If W@%30@_@%31@_@%28@_@%29@_@%02@ = CNS_NOT_KUH_Integer Then
      GoTo BOX_2
    End If
    GoTo BOX_E
  BOX_2:
Rem PRVLG
    @%16@
  BOX_3:
Rem PRVLG
    If W@%30@_@%31@_@%28@_@%29@_@%02@_wk <> CNS_NOT_KUH_Integer Then
      GoTo BOX_4
    End If
    GoTo BOX_5
  BOX_4:
Rem PRVLG
    W@%30@_@%31@_@%28@_@%29@_@%02@ = W@%30@_@%31@_@%28@_@%29@_@%02@_wk
Rem PRVLG
    CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
    GoTo BOX_E
  BOX_5:
```

## 【図 2 0 4】

付録一全ベクトルの型: 0 4 5 - 2

正規 (K の派生元) / 出力 / 配列無 / 等価無 / 境界無の L 3 (管理番号 0 5 1 0 - 2)

```
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 205】

付録一全ベクトルの型: 046-1

正規 (Kの派生元) / 出力 / 配列無 / 等価無 / 境界無の L4 (管理番号 0520-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
  Private W0%300_@%310_@%280_@%290_@%020 As @%230
Rem PUBIN4
  Private W0%3050_@%3150_@%2850_@%2950_@%2000 As @%2350
Rem PUB5C4
  Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
  Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
  Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
  Private CTRL_W0%300_@%310_@%280_@%290_@%020_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem PUB4B4
  ' Private W0%3000_@%3100_@%2800_@%2900_@%2000 As @%230
Rem PUB4N3
  Private W3_@%310_@%280_@%290_@%020 As Integer
Rem PUB4N2
  ' Private W2_@%310_@%280_@%290_@%020 As @%230
Rem #DEFPRV
Rem PRV2T4
  Private W0%300_@%310_@%280_@%290_@%020_wk As @%230
Rem PRV7R4
  Private CTRL_W0%300_@%290_ING_FLG As Boolean
Rem *PUBKU
  Private CNS_NOT_KUH_String As String
  Private CNS_NOT_KUH_Integer As Integer
  Private CNS_NOT_KUH_Boolean As Boolean
  Private CNS_NOT_KUH_Long As Long
  Private CNS_NOT_KUH_Single As Single
  Private CNS_NOT_KUH_Double As Double
  Private CNS_NOT_KUH_Variant As Variant
  Private CNS_NOT_KUH_Currency As Currency
  Private CNS_NOT_KUH_Byte As Byte
  Private CNS_NOT_KUH_Date As Date
Rem $PRV2W4
@%1720
Rem $PRV2E4
Rem $PRV4W4
@%1820
Rem $PRV4E4
Rem #DEFEND
  Public Sub Main0
Rem #VECREP Private Sub L4_@%310_@%280_@%290_@%0200
  BOX_1:
Rem PRVLG
    If W0%300_@%310_@%280_@%290_@%020_wk = CNS_NOT_KUH_@%230 And _
Rem PRVLG
    W3_@%310_@%280_@%290_@%020 = @%070 Then
      GoTo BOX_2
    End If
```

## 【図 2 0 6】

付録－全ベクトルの型： 0 4 6 - 2

正規（Kの派生元）／出力／配列無／等価無／境界無のL 4（管理番号 0 5 2 0 - 2）

```
      GoTo BOX_E
BOX_2:
Rem PUBIN
      If W@%305@_@%315@_@%285@_@%295@_@%200@ = CNS_NOT_KUH_@%235@ Then Go To Box_3
Rem PRVLG
      W@%30@_@%31@_@%28@_@%29@_@%02@_wk = W@%305@_@%315@_@%285@_@%295@_@%200@
BOX_3:
Rem PRVLG
      If W@%30@_@%31@_@%28@_@%29@_@%02@_wk <> CNS_NOT_KUH_@%23@ Then
          GoTo BOX_4
      End If
          GoTo BOX_5
BOX_4:
Rem PRVLG
      W@%30@_@%31@_@%28@_@%29@_@%02@ = W@%30@_@%31@_@%28@_@%29@_@%02@_wk
Rem PRVLG
      CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
          GoTo BOX_E
BOX_5:
Rem PRVLG
      If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
        = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
          GoTo BOX_6
      End If
          GoTo BOX_7
BOX_6:
Rem PRVLG
      CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
          GoTo BOX_E
BOX_7:
Rem PRVLG
      CTRL_W@%30@_@%29@_ING_FLG = True
          GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 2 0 7】

付録-全ベクトルの型: 0 4 7 - 1

正規 (K の派生元) / 出力 / 配列有 / 等価無 / 境界無の L 3 (管理番号 0 5 3 0 - 1)

```
Option Explicit
Rem #DEF PUB
Rem PUB4N3
Private %300_0%310_0%280_0%290_0%020_0%260_0%270 As Integer
Rem PUB1N3
Private %3050_0%3150_0%2850_0%2950_0%2000_0%2650_0%2750 As %2350
Rem PUB5C3
Private CTRL_%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG_0%260_0%270 As Boolean
Rem #DEFEND
Rem #DEF PRV
Rem PRV2T3
Private %300_0%310_0%280_0%290_0%020_0%260_0%270 As Integer
Rem PRV7R3
Private CTRL_%300_0%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_Integer As Integer
Private Gyou As Integer
Private Retu As Integer
Private Uti_Cnt As Integer
Private Soto_Cnt As Integer
Private Uti_Max As Integer
Private Soto_Max As Integer
Rem $PRV2W3
0%1620
Rem $PRV2E3
Rem $PRV4W3
0%1650
Rem $PRV4E3
Rem #DEFEND
Public Sub Main()
Rem #VECREP Private Sub L3_0%310_0%280_0%290_0%020()
    Gyou = 0
    Retu = 0
Rem PRVLG
    If 0%340 < 0%350 then
Rem PRVLG
        Uti_Max = 0%260
Rem PRVLG
        Soto_Max = 0%270
    End If
Rem PRVLG
    If 0%340 > 0%350 then
Rem PRVLG
        Uti_Max = 0%270
Rem PRVLG
        Soto_Max = 0%260
    End If
```

## 【図 208】

付録-全ベクトルの型: 047-2

正規 (Kの派生元) / 出力 / 配列有 / 等価無 / 境界無の L3 (管理番号 0530-2)

```
Uti_Cnt = 0
Soto_Cnt = 0
Do Until Soto_Cnt = Soto_Max + 1
  Uti_Cnt = 0
Rem PRVLG
  If %340 < %350 then
    Gyou = Uti_Cnt
    Retu = Soto_Cnt End If
Rem PRVLG
  If %340 > %350 then
    Gyou = Soto_Cnt
    Retu = Uti_Cnt End If
    Do Until Uti_Cnt = Uti_Max + 1
      BOX_1:
      Rem PRVLG
      If %300_%310_%280_%290_%020 (Gyou, Retu) = CNS_NOT_KUH_Integer Then
        GoTo BOX_2
      End If
      GoTo BOX_E
    BOX_2:
    Rem PRVLG
    %160
    BOX_3:
    Rem PRVLG
    If %300_%310_%280_%290_%020_wk (Gyou, Retu) <> CNS_NOT_KUH_Integer
      Then GoTo BOX_4
    End If
    GoTo BOX_5
    BOX_4:
    Rem PRVLG
    %300_%310_%280_%290_%020 (Gyou, Retu) = %300_%310_%280_%290_%020_wk (Gyou, Retu)
    Rem PRVLG
    CTRL_%300_%280_%290_STS_TRANSITION_FLG = CTRL_%300_%280_%290_STS_TRANSITION_FLG + 1
    GoTo BOX_E
    BOX_5:
    Rem PRVLG
    If CTRL_%300_%280_%290_STS_TRANSITION_FLG_P
      = CTRL_%300_%280_%290_STS_TRANSITION_FLG_PP Then GoTo BOX_6 End If GoTo BOX_7
    BOX_6:
    Rem PRVLG
    CTRL_%300_%310_%280_%290_%020_NOT_VALID_FLG (Gyou, Retu) = True
    GoTo BOX_E
    BOX_7:
    Rem PRVLG
    CTRL_%300_%290_ING_FLG = True GoTo BOX_E
    BOX_E:
    Uti_Cnt = Uti_Cnt + 1
    Rem PRVLG
    If %340 < %350 then
      Gyou = Uti_Cnt
      Retu = Soto_Cnt End If
    Rem PRVLG
    If %340 > %350 then
```

## 【図 2 0 9】

付録－全ベクトルの型： 0 4 7 - 3

正規（Kの派生元）／出力／配列有／等価無／境界無のL 3（管理番号 0 5 3 0 - 3）

---

```
        Gyou = Soto_Cnt
        Retu = Uti_Cnt  End If

    Loop
    Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
    If @@%34@ < @@%35@ then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt End If
Rem PRVLG
    If @@%34@ > @@%35@ then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt End If

    Loop
    Gyou = 0
    Retu = 0
    Uti_Cnt = 0
    Soto_Cnt = 0
    Uti_Max = 0
    Soto_Max = 0
End Sub
Rem *** PRCEND
```

---



## 【図 2 1 0】

付録-全ベクトルの型: 0 4 8 - 1

正規 (K の派生元) / 出力 / 配列有 / 等価無 / 境界無の L 4 (管理番号 0 5 4 0 - 1)

```

Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W0%300_0%310_0%280_0%290_0%020 (0%260, 0%270) As 0%230
Rem PUBIN4
Private W0%3050_0%3150_0%2850_0%2950_0%2000 (0%2650, 0%2750) As 0%2350
Rem PUB5C4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
Private CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG (0%260, 0%270) As Boolean
Rem #DEFEND
Rem PUB4B4
Private W0%3000_0%3100_0%2800_0%2900_0%2000 (0%260, 0%270) As 0%230
Rem PUB4N3
Private W3_0%310_0%280_0%290_0%020 (0%260, 0%270) As Integer
Rem PUB4N2
Private W2_0%310_0%280_0%290_0%020 (0%260, 0%270) As 0%230
Rem #DEFPRV
Rem PRV2T4
Private W0%300_0%310_0%280_0%290_0%020_wk (0%260, 0%270) As 0%230
Rem PRV7R4
Private CTRL_W0%300_0%290_INC_FLG As Boolean
Rem #PUBKU
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Private Gyou As Integer
Private Retu As Integer
Private Uti_Cnt As Integer
Private Soto_Cnt As Integer
Private Uti_Max As Integer
Private Soto_Max As Integer
Rem $PRV2W4
0%1720
Rem $PRV2E4
Rem $PRV4W4
0%1820
Rem $PRV4E4
Rem #DEFEND
Public Sub Main 0
Rem #VECREP Private Sub L4_0%310_0%280_0%290_0%020 0
Gyou = 0

```

## 【図 2 1 1】

付録-全ベクトルの型: 0 4 8-2

正規 (Kの派生元) / 出力 / 配列有 / 等価無 / 境界無の L 4 (管理番号 0 5 4 0-2)

```

Retu = 0
Rem PRVLG
  If 0%340 < 0%350 then
Rem PRVLG
  Uti_Max = 0%260
Rem PRVLG
  Soto_Max = 0%270
  End If
Rem PRVLG
  If 0%340 > 0%350 then
Rem PRVLG
  Uti_Max = 0%270
Rem PRVLG
  Soto_Max = 0%260
  End If
  Uti_Cnt = 0
  Soto_Cnt = 0
  Do Until Soto_Cnt = Soto_Max + 1
    Uti_Cnt = 0
Rem PRVLG
    If 0%340 < 0%350 then
      Gyou = Uti_Cnt
      Retu = Soto_Cnt
    End If
Rem PRVLG
    If 0%340 > 0%350 then
      Gyou = Soto_Cnt
      Retu = Uti_Cnt
    End If
    Do Until Uti_Cnt = Uti_Max + 1
BOX_1:
Rem PRVLG
  If W0%300_0%310_0%280_0%290_0%020_wk (Gyou, Retu) = CNS_NOT_KUH_0%230 And _
Rem PRVLG
  W3_0%310_0%280_0%290_0%020 (Gyou, Retu) = 0%070 Then
    GoTo BOX_2
  End If
    GoTo BOX_E
BOX_2:
Rem PUBIN
  If W0%3050_0%3150_0%2850_0%2950_0%2000 (Gyou, Retu) = CNS_NOT_KUH_0%2350 Then Go To Box_3
Rem PRVLG
  W0%300_0%310_0%280_0%290_0%020_wk (Gyou, Retu) = W0%3050_0%3150_0%2850_0%2950_0%2000 (Gyou, Retu)
BOX_3:
Rem PRVLG
  If W0%300_0%310_0%280_0%290_0%020_wk (Gyou, Retu) < CNS_NOT_KUH_0%230 Then
    GoTo BOX_4
  End If
    GoTo BOX_5
BOX_4:
Rem PRVLG
  W0%300_0%310_0%280_0%290_0%020 (Gyou, Retu) = W0%300_0%310_0%280_0%290_0%020_wk (Gyou, Retu)
Rem PRVLG

```

## 【図 212】

付録一全ベクトルの型: 048-3

正規 (Kの派生元) / 出力 / 配列有 / 等価無 / 境界無の L4 (管理番号 0540-3)

```
CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (Gyou, Retu) = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
  Uti_Cnt = Uti_Cnt + 1
Rem PRVLG
  If @%34@ < @%35@ then
    Gyou = Uti_Cnt
    Retu = Soto_Cnt
  End If
Rem PRVLG
  If @%34@ > @%35@ then
    Gyou = Soto_Cnt
    Retu = Uti_Cnt
  End If
  Loop
  Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
  If @%34@ < @%35@ then
    Gyou = Uti_Cnt
    Retu = Soto_Cnt
  End If
Rem PRVLG
  If @%34@ > @%35@ then
    Gyou = Soto_Cnt
    Retu = Uti_Cnt
  End If
  Loop
  Gyou = 0
  Retu = 0
  Uti_Cnt = 0
  Soto_Cnt = 0
  Uti_Max = 0
  Soto_Max = 0
End Sub
Rem *** PRCEND
```

## 【図 2 1 3】

付録ー全ベクトルの型: 0 4 9 - 1

正規 (K の派生元) / 出力 / 配列無 / 等価有 / 境界無の L 3 (管理番号 0 5 5 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
  Private W%300_@%310_@%280_@%290_@%020 As Integer
Rem PUBIN3
  Private W%3050_@%3150_@%2850_@%2950_@%2000 As @%2350
Rem PUB5C3
  Private CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
  Private CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
  Private CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
  Private CTRL_W%300_@%310_@%280_@%290_@%020_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
  Private W%300_@%310_@%280_@%290_@%020_wk%070 As Integer
Rem PRV7R3
  Private CTRL_W%300_@%290_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_Integer As Integer
Rem $PRV2W3
  @%1620
Rem $PRV2E3
Rem $PRV4W3
  @%1650
Rem $PRV4E3
Rem #DEFEND
  Public Sub Main()
Rem #VECREP Private Sub L3_@%310_@%280_@%290_@%020()
  BOX_1:
Rem PRVLG
  If W%300_@%310_@%280_@%290_@%020 = CNS_NOT_KUH_Integer Then
    GoTo BOX_2
  End If
  GoTo BOX_E
  BOX_2:
Rem PRVLG
  @%160
  BOX_3:
Rem PUBL3F
  If W%300_@%310_@%280_@%290_@%020_wk%070 = 1 Then GoTo BOX_4
  GoTo BOX_5
  BOX_4:
Rem PUBL3F
  If W%300_@%310_@%280_@%290_@%020_wk%070 = 1 Then W%300_@%310_@%280_@%290_@%020 = @%070
Rem PRVLG
  CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG = CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG + 1
  GoTo BOX_E
  BOX_5:
Rem PRVLG
  If CTRL_W%300_@%280_@%290_STS_TRANSITION_FLG_P
```

## 【図 2 1 4】

付録－全ベクトルの型： 0 4 9 - 2

正規（Kの派生元）／出力／配列無／等価有／境界無のL 3（管理番号 0 5 5 0 - 2）

```
= CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
  GoTo BOX_6
End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 2 1 5】

付録-全ベクトルの型 : 0 5 0 - 1

正規 (Kの派生元) / 出力 / 配列無 / 等価有 / 境界無の L4 (管理番号 0 5 6 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W0%300_0%310_0%280_0%290_0%020 As 0%230
Rem PUBIN4
Private W0%3050_0%3150_0%2850_0%2950_0%2000 As 0%2350
Rem PUB5C4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
Private CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem PUB4B4
Private W0%3000_0%3100_0%2800_0%2900_0%2000 As 0%230
Rem PUB4N3
Private W3_0%310_0%280_0%290_0%020 As Integer
Rem PUB4N2
Private W2_0%310_0%280_0%290_0%020 As 0%230
Rem #DEFPRV
Rem PRV2T4
Private W0%300_0%310_0%280_0%290_0%020_wk As 0%230
Rem PRV7R4
Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem #PUBKU
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Rem $PRV2W4
0%1720
Rem $PRV2E4
Rem $PRV4W4
0%1820
Rem $PRV4E4
Rem #DEFEND
Public Sub Main ()
Rem #VECREP Private Sub L4_0%310_0%280_0%290_0%020_0%070 ()
BOX_1:
Rem PRVLG
If W0%300_0%310_0%280_0%290_0%020_wk = CNS_NOT_KUH_0%230 And _
Rem PRVLG
W3_0%310_0%280_0%290_0%020 = 0%070 Then
GoTo BOX_2
End If
```

## 【図 2 1 6】

付録一全ベクトルの型: 0 5 0 - 2

正規 (K の派生元) / 出力 / 配列無 / 等価有 / 境界無の L 4 (管理番号 0 5 6 0 - 2)

```
      GoTo BOX_E
BOX_2:
Rem PUBIN
  If W@%305@_@%315@_@%285@_@%295@_@%200@ = CNS_NOT_KUH_@%235@ Then Go To Box_3
Rem PRVLG
  W@%30@_@%31@_@%28@_@%29@_@%02@_wk = W@%305@_@%315@_@%285@_@%295@_@%200@
BOX_3:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@_wk <> CNS_NOT_KUH_@%23@ Then
    GoTo BOX_4
  End If
  GoTo BOX_5
BOX_4:
Rem PRVLG
  W@%30@_@%31@_@%28@_@%29@_@%02@ = W@%30@_@%31@_@%28@_@%29@_@%02@_wk
Rem PRVLG
  CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 217】

付録-全ベクトルの型: 051-1

正規 (Kの派生元) / 出力 / 配列有 / 等価有 / 境界無の L3 (管理番号 0570-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
Private W%30%_%31%_%28%_%29%_%02% (%26%, %27%) As Integer
Rem PUBIN3
Private W%305%_%315%_%285%_%295%_%20% (%265%, %275%) As %235%
Rem PUB5C3
Private CTRL_W%30%_%28%_%29%_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_W%30%_%28%_%29%_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_W%30%_%28%_%29%_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_W%30%_%31%_%28%_%29%_%02%_NOT_VALID_FLG (%26%, %27%) As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
Private W%30%_%31%_%28%_%29%_%02%_wk%07% (%26%, %27%) As Integer
Rem PRV7R3
Private CTRL_W%30%_%29%_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH Integer As Integer
Private Gyou As Integer
Private Retu As Integer
Private Uti_Cnt As Integer
Private Soto_Cnt As Integer
Private Uti_Max As Integer
Private Soto_Max As Integer
Rem $PRV2W3
%162%
Rem $PRV2E3
Rem $PRV4W3
%165%
Rem $PRV4E3
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub L3_%31%_%28%_%29%_%02% 0
    Gyou = 0
    Retu = 0
Rem PRVLG
    If %34% < %35% then
Rem PRVLG
        Uti_Max = %26%
Rem PRVLG
        Soto_Max = %27%
    End If
Rem PRVLG
    If %34% > %35% then
Rem PRVLG
        Uti_Max = %27%
Rem PRVLG
        Soto_Max = %26%
    End If
```



## 【図 2 1 8】

付録－全ベクトルの型: 0 5 1 - 2

正規 (K の派生元) / 出力 / 配列有 / 等価有 / 境界無の L 3 (管理番号 0 5 7 0 - 2)

```
    Uti_Cnt = 0
    Soto_Cnt = 0
    Do Until Soto_Cnt = Soto_Max + 1
        Uti_Cnt = 0
    Rem PRVLG
        If 0%340 < 0%350 then
            Gyou = Uti_Cnt
            Retu = Soto_Cnt
        End If
    Rem PRVLG
        If 0%340 > 0%350 then
            Gyou = Soto_Cnt
            Retu = Uti_Cnt
        End If
    Do Until Uti_Cnt = Uti_Max + 1
    BOX_1:
    Rem PRVLG
        If W0%300_0%310_0%280_0%290_0%020 (Gyou, Retu) = CNS_NOT_KUH_Integer Then
            GoTo BOX_2
        End If
        GoTo BOX_E
    BOX_2:
    Rem PRVLG
        0%160
    BOX_3:
    Rem PUBL3F
        If W0%300_0%310_0%280_0%290_0%020_wk0%070 (Gyou, Retu) = 1 Then GoTo BOX_4
        GoTo BOX_5
    BOX_4:
    Rem PUBL3F
        If W0%300_0%310_0%280_0%290_0%020_wk0%070 (Gyou, Retu) = 1
            Then W0%300_0%310_0%280_0%290_0%020 (Gyou, Retu) = 0%070
    Rem PRVLG
        CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG = CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG + 1
        GoTo BOX_E
    BOX_5:
    Rem PRVLG
        If CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P
            = CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP Then
            GoTo BOX_6
        End If
        GoTo BOX_7
    BOX_6:
    Rem PRVLG
        CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG (Gyou, Retu) = True
        GoTo BOX_E
    BOX_7:
    Rem PRVLG
        CTRL_W0%300_0%290_ING_FLG = True
        GoTo BOX_E
    BOX_E:
        Uti_Cnt = Uti_Cnt + 1
    Rem PRVLG
```

## 【図 2 1 9】

付録一全ベクトルの型： 0 5 1 - 3

正規 (K の派生元) / 出力 / 配列有 / 等価有 / 境界無の L 3 (管理番号 0 5 7 0 - 3)

```
      If 0%340 < 0%350 then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
      End If
Rem PRVLG
      If 0%340 > 0%350 then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
      End If
      Loop
      Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
      If 0%340 < 0%350 then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
      End If
Rem PRVLG
      If 0%340 > 0%350 then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
      End If
      Loop
      Gyou = 0
      Retu = 0
      Uti_Cnt = 0
      Soto_Cnt = 0
      Uti_Max = 0
      Soto_Max = 0
End Sub
Rem *** PRCEND
```

## 【図 220】

付録一全ベクトルの型: 052-1

正規 (Kの派生元) / 出力 / 配列有 / 等価有 / 境界無のL4 (管理番号0580-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private #0%300_0%310_0%280_0%290_0%020_0%260_0%270 As 0%230
Rem PUBIN4
Private #0%3050_0%3150_0%2850_0%2950_0%2000_0%2650_0%2750 As 0%2350
Rem PUB5C4
Private CTRL_#0%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
Private CTRL_#0%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
Private CTRL_#0%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
Private CTRL_#0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG_0%260_0%270 As Boolean
Rem #DEFEND
Rem PUB4B4
Private #0%3000_0%3100_0%2800_0%2900_0%2000_0%260_0%270 As 0%230
Rem PUB4N3
Private #3_0%310_0%280_0%290_0%020_0%260_0%270 As Integer
Rem PUB4N2
Private #2_0%310_0%280_0%290_0%020_0%260_0%270 As 0%230
Rem #DEFPRV
Rem PRV2T4
Private #0%300_0%310_0%280_0%290_0%020_wk_0%260_0%270 As 0%230
Rem PRV7R4
Private CTRL_#0%300_0%290_ING_FLG As Boolean
Rem #PUBKU
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Private Gyou As Integer
Private Retu As Integer
Private Uti_Cnt As Integer
Private Soto_Cnt As Integer
Private Uti_Max As Integer
Private Soto_Max As Integer
Rem $PRV2W4
0%1720
Rem $PRV2E4
Rem $PRV4W4
0%1820
Rem $PRV4E4
Rem #DEFEND
Public Sub Main()
Rem #VECREP Private Sub L4_0%310_0%280_0%290_0%020_0%070 0
Gyou = 0
```

## 【図 2 2 1】

付録—全ベクトルの型: 0 5 2 - 2

正規 (Kの派生元) / 出力/配列有/等価有/境界無の L 4 (管理番号 0 5 8 0 - 2)

```
Retu = 0
Rem PRVLG
  If 0%346 < 0%350 then
Rem PRVLG
  Uti_Max = 0%260
Rem PRVLG
  Soto_Max = 0%270
  End If
Rem PRVLG
  If 0%340 > 0%350 then
Rem PRVLG
  Uti_Max = 0%270
Rem PRVLG
  Soto_Max = 0%260
  End If
  Uti_Cnt = 0
  Soto_Cnt = 0
  Do Until Soto_Cnt = Soto_Max + 1
    Uti_Cnt = 0
Rem PRVLG
    If 0%340 < 0%350 then
      Gyou = Uti_Cnt
      Retu = Soto_Cnt
    End If
Rem PRVLG
    If 0%340 > 0%350 then
      Gyou = Soto_Cnt
      Retu = Uti_Cnt
    End If
    Do Until Uti_Cnt = Uti_Max + 1
BOX_1:
Rem PRVLG
  If W0%300_0%310_0%280_0%290_0%020_wk (Gyou, Retu) = CNS_NOT_KUH_0%230 And _
Rem PRVLG
  W3_0%310_0%280_0%290_0%020 (Gyou, Retu) = 0%070 Then
    GoTo BOX_2
  End If
    GoTo BOX_E
BOX_2:
Rem PUBIN
  If W0%3050_0%3150_0%2850_0%2950_0%2000 (Gyou, Retu) = CNS_NOT_KUH_0%2350 Then Go To Box_3
Rem PRVLG
  W0%300_0%310_0%280_0%290_0%020_wk (Gyou, Retu) = W0%3050_0%3150_0%2850_0%2950_0%2000 (Gyou, Retu)
BOX_3:
Rem PRVLG
  If W0%300_0%310_0%280_0%290_0%020_wk (Gyou, Retu) <> CNS_NOT_KUH_0%230 Then
    GoTo BOX_4
  End If
    GoTo BOX_5
BOX_4:
Rem PRVLG
  W0%300_0%310_0%280_0%290_0%020 (Gyou, Retu) = W0%300_0%310_0%280_0%290_0%020_wk (Gyou, Retu)
Rem PRVLG
```

## 【図 2 2 2】

付録—全ベクトルの型 : 0 5 2 - 3

正規 (K の派生元) / 出力 / 配列有 / 等価有 / 境界無の L 4 (管理番号 0 5 8 0 - 3)

```

CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG = CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG + 1
GoTo BOX_E

BOX_5:
Rem PRVLG
If CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P
= CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP Then
GoTo BOX_6
End If
GoTo BOX_7

BOX_6:
Rem PRVLG
CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG (Gyou, Retu) = True
GoTo BOX_E

BOX_7:
Rem PRVLG
CTRL_W0%300_0%290_ING_FLG = True
GoTo BOX_E

BOX_E:
Uti_Cnt = Uti_Cnt + 1
Rem PRVLG
If 0%340 < 0%350 then
Gyou = Uti_Cnt
Retu = Soto_Cnt
End If
Rem PRVLG
If 0%340 > 0%350 then
Gyou = Soto_Cnt
Retu = Uti_Cnt
End If
Loop
Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
If 0%340 < 0%350 then
Gyou = Uti_Cnt
Retu = Soto_Cnt
End If
Rem PRVLG
If 0%340 > 0%350 then
Gyou = Soto_Cnt
Retu = Uti_Cnt
End If
Loop
Gyou = 0
Retu = 0
Uti_Cnt = 0
Soto_Cnt = 0
Uti_Max = 0
Soto_Max = 0
End Sub
Rem *** PRCEND

```

## 【図 223】

付録一全ベクトルの型: 053-1

正規 (Kの派生元) / 出力/配列無/等価無/境界有のL3 (管理番号0590-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
Private W0%300_0%310_0%280_0%290_0%020 As Integer
Rem PUBIN3
Private W0%3050_0%3150_0%2850_0%2950_0%2000 As 0%2350
Rem PUB5C3
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
Private W0%300_0%310_0%280_0%290_0%020_wk As Integer
Rem PRV7E3
Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_Integer As Integer
Rem $PRV2W3
0%1620
Rem $PRV2E3
Rem $PRV4W3
0%1650
Rem $PRV4E3
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub L3_0%310_0%280_0%290_0%0200
BOX_1:
Rem PRVLG
If W0%300_0%310_0%280_0%290_0%020 = CNS_NOT_KUH_Integer Then
GoTo BOX_2
End If
GoTo BOX_E
BOX_2:
Rem PRVLG
0%160
BOX_3:
Rem PRVLG
If W0%300_0%310_0%280_0%290_0%020_wk < CNS_NOT_KUH_Integer Then
GoTo BOX_4
End If
GoTo BOX_5
BOX_4:
Rem PRVLG
W0%300_0%310_0%280_0%290_0%020 = W0%300_0%310_0%280_0%290_0%020_wk
Rem PRVLG
CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG = CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG + 1
GoTo BOX_E
BOX_5:
```

## 【図 2 2 4】

付録－全ベクトルの型：0 5 3－2

正規（Kの派生元）／出力／配列無／等価無／境界有のL 3（管理番号0 5 9 0－2）

---

```
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

---

## 【図 2 2 5】

付録一全ベクトルの型: 0 5 4 - 1

正規 (K の派生元) / 出力 / 配列無 / 等価無 / 境界有の L 4 (管理番号 0 6 0 0 - 1)

```

Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W0%300_@%310_@%280_@%290_@%020 As @%230
Rem PUBIN4
Private W0%3050_@%3150_@%2850_@%2950_@%2000 As @%2350
Rem PUB5C4
Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
Private CTRL_W0%300_@%280_@%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
Private CTRL_W0%300_@%310_@%280_@%290_@%020_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem PUB4B4
' Private W0%3000_@%3100_@%2800_@%2900_@%2000 As @%230
Rem PUB4N3
Private W3_@%310_@%280_@%290_@%020 As Integer
Rem PUB4N2
' Private W2_@%310_@%280_@%290_@%020 As @%230
Rem #DEFPRV
Rem PRV2T4
Private W0%300_@%310_@%280_@%290_@%020_wk As @%230
Rem PRV7R4
Private CTRL_W0%300_@%290_ING_FLG As Boolean
Rem *PUBKU
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Rem $PRV2W4
@%1720
Rem $PRV2E4
Rem $PRV4W4
@%1820
Rem $PRV4E4
Rem #DEFEND
Public Sub Main 0
Rem #VECREP Private Sub L4_@%310_@%280_@%290_@%020 0
BOX_1:
Rem PRVLG
If W0%300_@%310_@%280_@%290_@%020 = CNS_NOT_KUH_@%230 And _
Rem PRVLG
W3_@%310_@%280_@%290_@%020 = @%070 Then
GoTo BOX_2
End If

```



## 【図 2 2 6】

付録一全ベクトルの型: 0 5 4 - 2

正規 (K の派生元) / 出力 / 配列無 / 等価無 / 境界有の L 4 (管理番号 0 6 0 0 - 2)

```
GoTo BOX_E
BOX_2:
Rem PUBIN
  If W@%305@_%315@_%285@_%295@_%200@ = CNS_NOT_KUH_@%235@ Then Go To Box_3
Rem PRVLG
  W@%30@_%31@_%28@_%29@_%02@_wk = W@%305@_%315@_%285@_%295@_%200@
BOX_3:
Rem PRVLG
  If W@%30@_%31@_%28@_%29@_%02@_wk <> CNS_NOT_KUH_@%23@ Then
    GoTo BOX_4
  End If
  GoTo BOX_5
BOX_4:
Rem PRVLG
  W@%30@_%31@_%28@_%29@_%02@ = W@%30@_%31@_%28@_%29@_%02@_wk
Rem PUB4B4L
  W@%300@_%310@_%280@_%290@_%200@ = W@%30@_%31@_%28@_%29@_%02@
Rem PRVLG
  CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG = CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_%28@_%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_%31@_%28@_%29@_%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 2 2 7】

付録-全ベクトルの型: 0 5 5 - 1

正規 (K の派生元) / 出力 / 配列無 / 等価有 / 境界有の L 3 (管理番号 0 6 1 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
Private W%300_0%310_0%280_0%290_0%020 As Integer
Rem PUBIN3
Private W%3050_0%3150_0%2850_0%2950_0%2000 As 0%2350
Rem PUB5C3
Private CTRL_W%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_W%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_W%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_W%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
Private W%300_0%310_0%280_0%290_0%020_wk0%070 As Integer
Rem PRV7R3
Private CTRL_W%300_0%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_Integer As Integer
Rem $PRV2W3
0%1620
Rem $PRV2E3
Rem $PRV4W3
0%1650
Rem $PRV4E3
Rem #DEFEND
Public Sub Main 0
Rem #VECREP Private Sub L3_0%310_0%280_0%290_0%020 0
BOX_1:
Rem PRVLG
If W%300_0%310_0%280_0%290_0%020 = CNS_NOT_KUH_Integer Then
GoTo BOX_2
End If
GoTo BOX_E
BOX_2:
Rem PRVLG
0%160
Rem PRVLG
W%300_0%310_0%280_0%290_0%020_wk0%070 = 0%070
BOX_3:
Rem PRVLG
If W%300_0%310_0%280_0%290_0%020_wk0%070 <> CNS_NOT_KUH_0%230 Then
GoTo BOX_4
End If
GoTo BOX_5
BOX_4:
Rem PRVLG
W%300_0%310_0%280_0%290_0%020 = W%300_0%310_0%280_0%290_0%020_wk0%070
Rem PRVLG
CTRL_W%300_0%280_0%290_STS_TRANSITION_FLG = CTRL_W%300_0%280_0%290_STS_TRANSITION_FLG + 1
```

## 【図 2 2 8】

付録一全ベクトルの型: 0 5 5 - 2

正規 (K の派生元) / 出力 / 配列無 / 等価有 / 境界有の L 3 (管理番号 0 6 1 0 - 2)

```
      GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 2 2 9】

付録—全ベクトルの型: 0 5 6 - 1

正規 (K の派生元) / 出力 / 配列無 / 等価有 / 境界有の L 4 (管理番号 0 6 2 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W0%300_0%310_0%280_0%290_0%020 As 0%230
Rem PUB1N4
Private W0%3050_0%3150_0%2850_0%2950_0%2000 As 0%2350
Rem PUB5C4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
Private CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem PUB4B4
Private W0%3000_0%3100_0%2800_0%2900_0%2000 As 0%230
Rem PUB4N3
Private W3_0%310_0%280_0%290_0%020 As Integer
Rem PUB4N2
Private W2_0%310_0%280_0%290_0%020 As 0%230
Rem #DEFPRV
Rem PRV2T4
Private W0%300_0%310_0%280_0%290_0%020_wk As 0%230
Rem PRV7R4
Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem #PUBKU
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Rem $PRV2W4
0%1720
Rem $PRV2E4
Rem $PRV4W4
0%1820
Rem $PRV4E4
Rem #DEFEND
Public Sub Main 0
Rem #VECREP Private Sub L4_0%310_0%280_0%290_0%020_0%070 0
BOX_1:
Rem PRVLG
If W0%300_0%310_0%280_0%290_0%020 = CNS_NOT_KUH_0%230 And _
Rem PRVLG
W3_0%310_0%280_0%290_0%020 = 0%070 Then
GoTo BOX_2
End If
```

## 【図 2 3 0】

付録-全ベクトルの型: 0 5 6 - 2

正規 (K の派生元) / 出力 / 配列無 / 等価有 / 境界有の L 4 (管理番号 0 6 2 0 - 2)

```
GoTo BOX_E
BOX_2:
Rem PUBIN
  If W@%305@_@%315@_@%285@_@%295@_@%200@ = CNS_NOT_KUH_@%235@ Then Go To Box_3
Rem PRVLG
  W@%30@_@%31@_@%28@_@%29@_@%02@_wk = W@%305@_@%315@_@%285@_@%295@_@%200@
BOX_3:
Rem PRVLG
  If W@%30@_@%31@_@%28@_@%29@_@%02@_wk <> CNS_NOT_KUH_@%23@ Then
    GoTo BOX_4
  End If
  GoTo BOX_5
BOX_4:
Rem PRVLG
  W@%30@_@%31@_@%28@_@%29@_@%02@ = W@%30@_@%31@_@%28@_@%29@_@%02@_wk
Rem PUB4B4L
  W@%300@_@%310@_@%280@_@%290@_@%200@ = W@%30@_@%31@_@%28@_@%29@_@%02@
Rem PRVLG
  CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG + 1
  GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P
    = CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W@%30@_@%29@_ING_FLG = True
  GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 231】

付録一全ベクトルの型: 057-1

正規 (K の派生元) / 出力 / 配列有 / 等価無 / 境界有の L3 (管理番号 0630-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
Private W%300_0%310_0%280_0%290_0%020 (6%260, 0%270) As Integer
Rem PUB1N3
Private W%3050_0%3150_0%2850_0%2950_0%2000 (0%2650, 0%2750) As 0%2350
Rem PUB5C3
Private CTRL_W%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_W%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_W%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_W%300_0%310_6%280_0%290_0%020_NOT_VALID_FLG (0%260, 0%270) As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
Private W%300_0%310_6%280_0%290_0%020_wk (0%260, 0%270) As Integer
Rem PRV7R3
Private CTRL_W%300_0%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_Integer As Integer
Private Gyou As Integer
Private Retu As Integer
Private Uti_Cnt As Integer
Private Solo_Cnt As Integer
Private Uti_Max As Integer
Private Solo_Max As Integer
Rem $PRV2W3
0%1620
Rem $PRV2E3
Rem $PRV4W3
0%1650
Rem $PRV4E3
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub L3_0%310_0%280_0%290_0%020 0
    Gyou = 0
    Retu = 0
Rem PRVLG
    If 0%340 < 0%350 then
Rem PRVLG
        Uti_Max = 0%260
Rem PRVLG
        Solo_Max = 0%270
    End If
Rem PRVLG
    If 0%340 > 0%350 then
Rem PRVLG
        Uti_Max = 0%270
Rem PRVLG
        Solo_Max = 0%260
    End If
```

## 【図 2 3 2】

付録-全ベクトルの型: 0 5 7-2

正規 (Kの派生元) / 出力 / 配列有 / 等価無 / 境界有の L 3 (管理番号 0 6 3 0 - 2)

```

    Uti_Cnt = 0
    Solo_Cnt = 0
    Do Until Solo_Cnt = Solo_Max + 1
        Uti_Cnt = 0
    Rem PRVLG
        If 0%340 < 0%350 then
            Gyou = Uti_Cnt
            Retu = Solo_Cnt
        End If
    Rem PRVLG
        If 0%340 > 0%350 then
            Gyou = Solo_Cnt
            Retu = Uti_Cnt
        End If
        Do Until Uti_Cnt = Uti_Max + 1
    BOX_1:
    Rem PRVLG
        If W0%300_0%310_0%280_0%290_0%020 (Gyou, Retu) = CNS_NOT_KUH_Integer Then
            GoTo BOX_2
        End If
        GoTo BOX_E
    BOX_2:
    Rem PRVLG
        0%160
    BOX_3:
    Rem PUBL3P
        If W0%300_0%310_0%280_0%290_0%020_wk (Gyou, Retu) = 1 Then GoTo BOX_4
        GoTo BOX_5
    BOX_4:
    Rem PUBL3P
        If W0%300_0%310_0%280_0%290_0%020_wk (Gyou, Retu) = 1
            Then W0%300_0%310_0%280_0%290_0%020 (Gyou, Retu) = 0%070
    Rem PRVLG
        CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG = CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG + 1
        GoTo BOX_E
    BOX_5:
    Rem PRVLG
        If CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P
            = CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP Then
            GoTo BOX_6
        End If
        GoTo BOX_7
    BOX_6:
    Rem PRVLG
        CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG (Gyou, Retu) = True
        GoTo BOX_E
    BOX_7:
    Rem PRVLG
        CTRL_W0%300_0%290_ING_FLG = True
        GoTo BOX_E
    BOX_E:
        Uti_Cnt = Uti_Cnt + 1
    Rem PRVLG

```

## 【図 2 3 3】

付録一全ベクトルの型： 0 5 7 - 3

正規 (Kの派生元) / 出力 / 配列有 / 等価無 / 境界有の L 3 (管理番号 0 6 3 0 - 3)

---

```
        If @@%34@ < @@%35@ then
            Gyou = Uti_Cnt
            Retu = Soto_Cnt
        End If
Rem PRVLG
        If @@%34@ > @@%35@ then
            Gyou = Soto_Cnt
            Retu = Uti_Cnt
        End If
        Loop
        Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
        If @@%34@ < @@%35@ then
            Gyou = Uti_Cnt
            Retu = Soto_Cnt
        End If
Rem PRVLG
        If @@%34@ > @@%35@ then
            Gyou = Soto_Cnt
            Retu = Uti_Cnt
        End If
        Loop
        Gyou = 0
        Retu = 0
        Uti_Cnt = 0
        Soto_Cnt = 0
        Uti_Max = 0
        Soto_Max = 0
End Sub
Rem *** PRCEND
```

---



## 【図 2 3 4】

付録－全ベクトルの型 : 0 5 8 - 1

正規 (K の派生元) / 出力 / 配列有 / 等価無 / 境界有の L 4 (管理番号 0 6 4 0 - 1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
  Private W@%30@_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As @%23@
Rem PUBIN4
  Private W@%305@_@%315@_@%285@_@%295@_@%200@ (@%265@, @%275@) As @%235@
Rem PUB5C4
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
  Private CTRL_W@%30@_@%28@_@%29@_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
  Private CTRL_W@%30@_@%31@_@%28@_@%29@_@%02@_NOT_VALID_FLG (@%26@, @%27@) As Boolean
Rem #DEFEND
Rem PUB4B4
  ' Private W@%300@_@%310@_@%280@_@%290@_@%200@ (@%26@, @%27@) As @%23@
Rem PUB4N3
  Private W3_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As Integer
Rem PUB4N2
  ' Private W2_@%31@_@%28@_@%29@_@%02@ (@%26@, @%27@) As @%23@
Rem #DEFPRV
Rem PRV2T4
  Private W@%30@_@%31@_@%28@_@%29@_@%02@_wk (@%26@, @%27@) As @%23@
Rem PRV7R4
  Private CTRL_W@%30@_@%29@_ING_FLG As Boolean
Rem *PUBKU
  Private CNS_NOT_KUH_String As String
  Private CNS_NOT_KUH_Integer As Integer
  Private CNS_NOT_KUH_Boolean As Boolean
  Private CNS_NOT_KUH_Long As Long
  Private CNS_NOT_KUH_Single As Single
  Private CNS_NOT_KUH_Double As Double
  Private CNS_NOT_KUH_Variant As Variant
  Private CNS_NOT_KUH_Currency As Currency
  Private CNS_NOT_KUH_Byte As Byte
  Private CNS_NOT_KUH_Date As Date
  Private Gyou as Integer
  Private Retu as Integer
  Private Uti_Cnt as Integer
  Private Soto_Cnt as Integer
  Private Uti_Max as Integer
  Private Soto_Max as Integer
Rem $PRV2W4
@%172@
Rem $PRV2E4
Rem $PRV4W4
@%182@
Rem $PRV4E4
Rem #DEFEND
  Public Sub Main 0
Rem #VECREP Private Sub L4_@%31@_@%28@_@%29@_@%02@ 0
  Gyou = 0
```

## 【図 235】

付録-全ベクトルの型: 058-2

正規 (Kの派生元) / 出力/配列有/等価無/境界有のL4 (管理番号0640-2)

```
Retu = 0
Rem PRVLG
  If 0%340 < 0%350 then
Rem PRVLG
  Uti_Max = 0%260
Rem PRVLG
  Soto_Max = 0%270
  End If
Rem PRVLG
  If 0%340 > 0%350 then
Rem PRVLG
  Uti_Max = 0%270
Rem PRVLG
  Soto_Max = 0%260
  End If
  Uti_Cnt = 0
  Soto_Cnt = 0
  Do Until Soto_Cnt = Soto_Max + 1
    Uti_Cnt = 0
Rem PRVLG
    If 0%340 < 0%350 then
      Gyou = Uti_Cnt
      Retu = Soto_Cnt
    End If
Rem PRVLG
    If 0%340 > 0%350 then
      Gyou = Soto_Cnt
      Retu = Uti_Cnt
    End If
    Do Until Uti_Cnt = Uti_Max + 1
BOX_1:
Rem PRVLG
  If W0%300_0%310_0%280_0%290_0%020 (Gyou, Retu) = CNS_NOT_KUH_0%230 And _
Rem PRVLG
  W3_0%310_0%280_0%290_0%020 (Gyou, Retu) = 0%070 Then
    GoTo BOX_2
  End If
    GoTo BOX_E
BOX_2:
Rem PUBIN
  If W0%3050_0%3150_0%2850_0%2950_0%2000 (Gyou, Retu) = CNS_NOT_KUH_0%2350 Then Go To Box_3
Rem PRVLG
  W0%300_0%310_0%280_0%290_0%020_wk (Gyou, Retu) = W0%3050_0%3150_0%2850_0%2950_0%2000 (Gyou, Retu)
BOX_3:
Rem PRVLG
  If W0%300_0%310_0%280_0%290_0%020_wk (Gyou, Retu) < CNS_NOT_KUH_0%230 Then
    GoTo BOX_4
  End If
    GoTo BOX_5
BOX_4:
Rem PRVLG
  W0%300_0%310_0%280_0%290_0%020 (Gyou, Retu) = W0%300_0%310_0%280_0%290_0%020_wk (Gyou, Retu)
Rem PUB4B4L
```

## 【図 2 3 6】

付録-全ベクトルの型: 0 5 8 - 3

正規 (Kの派生元) / 出力 / 配列有 / 等価無 / 境界有の L 4 (管理番号 0 6 4 0 - 3)

```
%300_%310_%280_%290_%200 (Gyou, Retu) = %300_%310_%280_%290_%200 (Gyou, Retu)
Rem PRVLG
CTRL_%300_%280_%290_STS_TRANSITION_FLG = CTRL_%300_%280_%290_STS_TRANSITION_FLG + 1
GoTo BOX_E
BOX_5:
Rem PRVLG
If CTRL_%300_%280_%290_STS_TRANSITION_FLG_P
= CTRL_%300_%280_%290_STS_TRANSITION_FLG_PP Then
GoTo BOX_6
End If
GoTo BOX_7
BOX_6:
Rem PRVLG
CTRL_%300_%310_%280_%290_%200_NOT_VALID_FLG (Gyou, Retu) = True
GoTo BOX_E
BOX_7:
Rem PRVLG
CTRL_%300_%290_ING_FLG = True
GoTo BOX_E
BOX_E:
Uti_Cnt = Uti_Cnt + 1
Rem PRVLG
If %340 < %350 then
Gyou = Uti_Cnt
Retu = Soto_Cnt
End If
Rem PRVLG
If %340 > %350 then
Gyou = Soto_Cnt
Retu = Uti_Cnt
End If
Loop
Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
If %340 < %350 then
Gyou = Uti_Cnt
Retu = Soto_Cnt
End If
Rem PRVLG
If %340 > %350 then
Gyou = Soto_Cnt
Retu = Uti_Cnt
End If
Loop
Gyou = 0
Retu = 0
Uti_Cnt = 0
Soto_Cnt = 0
Uti_Max = 0
Soto_Max = 0
End Sub
Rem *** PRCEnd
```

## 【図 237】

付録一全ベクトルの型: 059-1

正規 (Kの派生元) / 出力 / 配列有 / 等価有 / 境界有の L3 (管理番号 0650-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N3
Private W0%300_0%310_0%280_0%290_0%020 (0%260, 0%270) As Integer
Rem PUBIN3
Private W0%3050_0%3150_0%2850_0%2950_0%2000 (0%2650, 0%2750) As 0%2350
Rem PUB5C3
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y3
Private CTRL_0%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B3
Private CTRL_0%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F3
Private CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG (0%260, 0%270) As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T3
Private W0%300_0%310_0%280_0%290_0%020_W0%070 (0%260, 0%270) As Integer
Rem PRV7R3
Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_Integer As Integer
Private Gyou as Integer
Private Retu as Integer
Private Uti_Cnt as Integer
Private Solo_Cnt as Integer
Private Uti_Max as Integer
Private Soto_Max as Integer
Rem $PRV243
0%1620
Rem $PRV2E3
Rem $PRV4W3
0%1650
Rem $PRV4E3
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub L3_0%310_0%280_0%290_0%020 0
    Gyou = 0
    Retu = 0
Rem PRVLG
    If 0%340 < 0%350 then
Rem PRVLG
        Uti_Max = 0%260
Rem PRVLG
        Soto_Max = 0%270
    End If
Rem PRVLG
    If 0%340 > 0%350 then
Rem PRVLG
        Uti_Max = 0%270
Rem PRVLG
        Soto_Max = 0%260
    End If
```

## 【図 2 3 8】

付録一全ベクトルの型: 0 5 9 - 2

正規 (K の派生元) / 出力 / 配列有 / 等価有 / 境界有の L 3 (管理番号 0 6 5 0 - 2)

```

    Uti_Cnt = 0
    Soto_Cnt = 0
    Do Until Soto_Cnt = Soto_Max + 1
        Uti_Cnt = 0
    Rem PRVLG
        If 0%340 < 0%350 then
            Gyou = Uti_Cnt
            Retu = Soto_Cnt
        End If
    Rem PRVLG
        If 0%340 > 0%350 then
            Gyou = Soto_Cnt
            Retu = Uti_Cnt
        End If
        Do Until Uti_Cnt = Uti_Max + 1
    BOX_1:
    Rem PRVLG
        If W0%300_0%310_0%280_0%290_0%020 (Gyou, Retu) = CNS_NOT_KUH_Integer Then
            GoTo BOX_2
        End If
        GoTo BOX_E
    BOX_2:
    Rem PRVLG
        0%160
    BOX_3:
    Rem PUBL3F
        If W0%300_0%310_0%280_0%290_0%020_wk0%070 (Gyou, Retu) = 1 Then GoTo BOX_4
        GoTo BOX_5
    BOX_4:
    Rem PUBL3F
        If W0%300_0%310_0%280_0%290_0%020_wk0%070 (Gyou, Retu) = 1
            Then W0%300_0%310_0%280_0%290_0%020 (Gyou, Retu) = 0%070
    Rem PRVLG
        CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG = CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG + 1
        GoTo BOX_E
    BOX_5:
    Rem PRVLG
        If CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P
            = CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP Then
            GoTo BOX_6
        End If
        GoTo BOX_7
    BOX_6:
    Rem PRVLG
        CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG (Gyou, Retu) = True
        GoTo BOX_E
    BOX_7:
    Rem PRVLG
        CTRL_W0%300_0%290_ING_FLG = True
        GoTo BOX_E
    BOX_E:
        Uti_Cnt = Uti_Cnt + 1
    Rem PRVLG

```

## 【図 239】

付録—全ベクトルの型: 059-3

正規 (Kの派生元) / 出力 / 配列有 / 等価有 / 境界有のL3 (管理番号0650-3)

```
      If 0%340 < 0%350 then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
      End If
Rem PRVLG
      If 0%340 > 0%350 then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
      End If
      Loop
      Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
      If 0%340 < 0%350 then
        Gyou = Uti_Cnt
        Retu = Soto_Cnt
      End If
Rem PRVLG
      If 0%340 > 0%350 then
        Gyou = Soto_Cnt
        Retu = Uti_Cnt
      End If
      Loop
      Gyou = 0
      Retu = 0
      Uti_Cnt = 0
      Soto_Cnt = 0
      Uti_Max = 0
      Soto_Max = 0
End Sub
Rem *** PRCEND
```

## 【図 240】

付録—全ベクトルの型: 060-1

正規 (Kの派生元) / 出力 / 配列有 / 等価有 / 境界有の L4 (管理番号 0660-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W0%300_0%310_0%280_0%290_0%020 (0%260, 0%270) As 0%230
Rem PUBIN4
Private W0%3050_0%3150_0%2850_0%2950_0%2000 (0%2650, 0%2750) As 0%2350
Rem PUB5C4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B4
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F4
Private CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG (0%260, 0%270) As Boolean
Rem #DEFEND
Rem PUB4B4
Private W0%3000_0%3100_0%2800_0%2900_0%2000 (0%260, 0%270) As 0%230
Rem PUB4N3
Private W3_0%310_0%280_0%290_0%020 (0%260, 0%270) As Integer
Rem PUB4N2
Private W2_0%310_0%280_0%290_0%020 (0%260, 0%270) As 0%230
Rem #DEFPRV
Rem PRV2T4
Private W0%300_0%310_0%280_0%290_0%020_wk (0%260, 0%270) As 0%230
Rem PRV7R4
Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem *PUBKU
Private CNS_NOT_KUH_String As String
Private CNS_NOT_KUH_Integer As Integer
Private CNS_NOT_KUH_Boolean As Boolean
Private CNS_NOT_KUH_Long As Long
Private CNS_NOT_KUH_Single As Single
Private CNS_NOT_KUH_Double As Double
Private CNS_NOT_KUH_Variant As Variant
Private CNS_NOT_KUH_Currency As Currency
Private CNS_NOT_KUH_Byte As Byte
Private CNS_NOT_KUH_Date As Date
Private Gyou As Integer
Private Retu As Integer
Private Uti_Cnt As Integer
Private Soto_Cnt As Integer
Private Uti_Max As Integer
Private Soto_Max As Integer
Rem $PRV2W4
0%1720
Rem $PRV2E4
Rem $PRV4W4
0%1820
Rem $PRV4E4
Rem #DEFEND
Public Sub Main 0
Rem #VECREP Private Sub L4_0%310_0%280_0%290_0%020_0%070 0
Gyou = 0
```

## 【図 2 4 1】

付録-全ベクトルの型: 0 6 0 - 2

正規 (Kの派生元) / 出力/配列有/等価有/境界有のL 4 (管理番号 0 6 6 0 - 2)

```
Retu = 0
Rem PRVLG
  If 0x340 < 0x350 then
Rem PRVLG
  Uti_Max = 0x260
Rem PRVLG
  Soto_Max = 0x270
  End If
Rem PRVLG
  If 0x340 > 0x350 then
Rem PRVLG
  Uti_Max = 0x270
Rem PRVLG
  Soto_Max = 0x260
  End If
  Uti_Cnt = 0
  Soto_Cnt = 0
  Do Until Soto_Cnt = Soto_Max + 1
    Uti_Cnt = 0
Rem PRVLG
    If 0x340 < 0x350 then
      Gyou = Uti_Cnt
      Retu = Soto_Cnt
    End If
Rem PRVLG
    If 0x340 > 0x350 then
      Gyou = Soto_Cnt
      Retu = Uti_Cnt
    End If
    Do Until Uti_Cnt = Uti_Max + 1
BOX_1:
Rem PRVLG
  If W0x300_0x310_0x280_0x290_0x020 (Gyou, Retu) = CNS_NOT_KUH_0x230 And _
Rem PRVLG
  W3_0x310_0x280_0x290_0x020 (Gyou, Retu) = 0x070 Then
    GoTo BOX_2
  End If
  GoTo BOX_E
BOX_2:
Rem PUBIN
  If W0x3050_0x3150_0x2850_0x2950_0x2000 (Gyou, Retu) = CNS_NOT_KUH_0x2350 Then Go To Box_3
Rem PRVLG
  W0x300_0x310_0x280_0x290_0x020_wk (Gyou, Retu) = W0x3050_0x3150_0x2850_0x2950_0x2000 (Gyou, Retu)
BOX_3:
Rem PRVLG
  If W0x300_0x310_0x280_0x290_0x020_wk (Gyou, Retu) <> CNS_NOT_KUH_0x230 Then
    GoTo BOX_4
  End If
  GoTo BOX_5
BOX_4:
Rem PRVLG
  W0x300_0x310_0x280_0x290_0x020 (Gyou, Retu) = W0x300_0x310_0x280_0x290_0x020_wk (Gyou, Retu)
Rem PUB4B4L
```



## 【図 2 4 2】

付録一全ベクトルの型: 0 6 0 - 3

正規 (K の派生元) / 出力 / 配列有 / 等価有 / 境界有の L 4 (管理番号 0 6 6 0 - 3)

```
W0%300_0%310_0%280_0%290_0%200 (Gyou, Retu) = W0%300_0%310_0%280_0%290_0%200 (Gyou, Retu)
Rem PRVLG
  CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG = CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG + 1
  GoTo BOX_E
BOX_5:
Rem PRVLG
  If CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P
    = CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP Then
    GoTo BOX_6
  End If
  GoTo BOX_7
BOX_6:
Rem PRVLG
  CTRL_W0%300_0%310_0%280_0%290_0%202_NOT_VALID_FLG (Gyou, Retu) = True
  GoTo BOX_E
BOX_7:
Rem PRVLG
  CTRL_W0%300_0%290_ING_FLG = True
  GoTo BOX_E
BOX_E:
  Uti_Cnt = Uti_Cnt + 1
Rem PRVLG
  If 0%340 < 0%350 then
    Gyou = Uti_Cnt
    Retu = Soto_Cnt
  End If
Rem PRVLG
  If 0%340 > 0%350 then
    Gyou = Soto_Cnt
    Retu = Uti_Cnt
  End If
  Loop
  Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
  If 0%340 < 0%350 then
    Gyou = Uti_Cnt
    Retu = Soto_Cnt
  End If
Rem PRVLG
  If 0%340 > 0%350 then
    Gyou = Soto_Cnt
    Retu = Uti_Cnt
  End If
  Loop
  Gyou = 0
  Retu = 0
  Uti_Cnt = 0
  Soto_Cnt = 0
  Uti_Max = 0
  Soto_Max = 0
End Sub
Rem *** PCEND
```

## 【図 243】

付録-全ベクトルの型: 061-1

K/入力/配列無/等価無/境界無のL2 (管理番号0670-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N2
Private W0%300_0%310_0%280_0%290_0%020 As 0%230
Rem PUB1N2
Private W0%3050_0%3150_0%2850_0%2950_0%2000 As 0%2350
Rem PUB5C2
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y2
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B2
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F2
Private CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG As Boolean
Rem #DEFEND
Rem #DEFFPRV
Rem PRV2T2
Private W0%300_0%310_0%280_0%290_0%020_wk As 0%230
Rem PRV7R2
Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_0%230 As 0%230
Rem $PRV2W2
0%1520
Rem $PRV2E2
Rem $PRV4W2
0%1550
Rem $PRV4E2
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub L2_0%310_0%280_0%290_0%020 0
BOX_1:
Rem PRVLG
If W0%300_0%310_0%280_0%290_0%020 = CNS_NOT_KUH_0%230 Then
GoTo BOX_2
End If
GoTo BOX_E
BOX_2:
' NOP
BOX_3:
Rem PRVLG
If W0%300_0%310_0%280_0%290_0%020_wk <> CNS_NOT_KUH_0%230 Then
GoTo BOX_4
End If
GoTo BOX_5
BOX_4:
Rem PRVLG
W0%300_0%310_0%280_0%290_0%020 = W0%300_0%310_0%280_0%290_0%020
Rem PRVLG
CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG = CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG + 1
GoTo BOX_E
BOX_5:
Rem PRVLG
```

## 【図 2 4 4】

付録一全ベクトルの型 : 0 6 1 - 2

K / 入力 / 配列無 / 等価無 / 境界無の L 2 (管理番号 0 6 7 0 - 2)

```

    If CTRL_W0x300_0x280_0x290_STS_TRANSITION_FLG_P
      = CTRL_W0x300_0x280_0x290_STS_TRANSITION_FLG_PP Then
        GoTo BOX_6
      End If
        GoTo BOX_7
BOX_6:
Rem PRVLG
    CTRL_W0x300_0x310_0x280_0x290_0x020_NOT_VALID_FLG = True
        GoTo BOX_E
BOX_7:
Rem PRVLG
    CTRL_W0x300_0x290_ING_FLG = True
        GoTo BOX_E
BOX_E:
End Sub
Rem *** PRCEND
```

## 【図 245】

付録一全ベクトルの型: 062-1

K/入力/配列有/等価無/境界無のL2 (管理番号0680-1)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N2
Private W0%300_0%310_0%280_0%290_0%020 (0%260, 0%270) As 0%230
Rem PUB1N2
Private W0%3050_0%3150_0%2850_0%2950_0%2000 As 0%2350
Rem PUB5C2
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG As Integer
Rem PUB5Y2
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P As Integer
Rem PUB5B2
Private CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP As Integer
Rem PUB6F2
Private CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG (0%260, 0%270) As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV2T2
Private W0%300_0%310_0%280_0%290_0%020_wk (0%260, 0%270) As 0%230
Rem PRV7R2
Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_0%230 As 0%230
Private Gyou as Integer
Private Retu as Integer
Private Uti_Cnt as Integer
Private Soto_Cnt as Integer
Private Uti_Max as Integer
Private Soto_Max as Integer
Rem $PRV2W2
0%1520
Rem $PRV2E2
Rem $PRV4W2
0%1550
Rem $PRV4E2
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub L2_0%310_0%280_0%290_0%0200
    Gyou = 0
    Retu = 0
Rem PRVLG
    If 0%340 < 0%350 then
Rem PRVLG
        Uti_Max = 0%260
Rem PRVLG
        Soto_Max = 0%270
    End If
Rem PRVLG
    If 0%340 > 0%350 then
Rem PRVLG
        Uti_Max = 0%270
Rem PRVLG
        Soto_Max = 0%260
    End If
```

## 【図 246】

付録-全ベクトルの型: 062-2

K/入力/配列有/等価無/境界無のL2 (管理番号0680-2)

```
      Uti_Cnt = 0
      Soto_Cnt = 0
      Do Until Soto_Cnt = Soto_Max + 1
        Uti_Cnt = 0
      Rem PRVLG
        If 0%340 < 0%350 then
          Gyou = Uti_Cnt
          Retu = Soto_Cnt
        End If
      Rem PRVLG
        If 0%340 > 0%350 then
          Gyou = Soto_Cnt
          Retu = Uti_Cnt
        End If
      Do Until Uti_Cnt = Uti_Max + 1
      BOX_1:
      Rem PRVLG
        If W0%300_0%310_0%280_0%290_0%020 (Gyou, Retu) = CNS_NOT_KUH_0%230 Then
          GoTo BOX_2
        End If
        GoTo BOX_E
      BOX_2:
        'NOP
      BOX_3:
      Rem PRVLG
        If W0%300_0%310_0%280_0%290_0%020_wk (Gyou, Retu) <> CNS_NOT_KUH_0%230 Then
          GoTo BOX_4
        End If
        GoTo BOX_5
      BOX_4:
      Rem PRVLG
        W0%300_0%310_0%280_0%290_0%020 (Gyou, Retu) = W0%300_0%310_0%280_0%290_0%020 (Gyou, Retu)
      Rem PRVLG
        CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG = CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG + 1
        GoTo BOX_E
      BOX_5:
      Rem PRVLG
        If CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_P
          = CTRL_W0%300_0%280_0%290_STS_TRANSITION_FLG_PP Then
          GoTo BOX_6
        End If
        GoTo BOX_7
      BOX_6:
      Rem PRVLG
        CTRL_W0%300_0%310_0%280_0%290_0%020_NOT_VALID_FLG (Gyou, Retu) = True
        GoTo BOX_E
      BOX_7:
      Rem PRVLG
        CTRL_W0%300_0%290_ING_FLG = True
        GoTo BOX_E
      BOX_E:
        Uti_Cnt = Uti_Cnt + 1
      Rem PRVLG
```

## 【図 247】

付録一全ベクトルの型: 062-3

K/入力/配列有/等価無/境界無のL2 (管理番号0680-3)

```

        If 0%340 < 0%350 then
            Gyou = Uti_Cnt
            Retu = Soto_Cnt
        End If
Rem PRVLG
        If 0%340 > 0%350 then
            Gyou = Soto_Cnt
            Retu = Uti_Cnt
        End If

        Loop
        Soto_Cnt = Soto_Cnt + 1
Rem PRVLG
        If 0%340 < 0%350 then
            Gyou = Uti_Cnt
            Retu = Soto_Cnt
        End If
Rem PRVLG
        If 0%340 > 0%350 then
            Gyou = Soto_Cnt
            Retu = Uti_Cnt
        End If

        Loop
        Gyou = 0
        Retu = 0
        Uti_Cnt = 0
        Soto_Cnt = 0
        Uti_Max = 0
        Soto_Max = 0
End Sub
Rem *** PRCEND

```

## 【図 248】

付録一全ベクトルの型: 063

I 2第2のS4 (管理番号0690)

```

Option Explicit
Rem #DEFPUB
Rem PUB4N4
    Private W0%300_0%310_0%280_0%290_0%020 As 0%230
Rem #DEFEND
Rem #DEFPVR
Rem PUBKU
    Private CNS_NOT_KUH_0%230 As 0%230
Rem #DEFEND
    Public Sub Main0
Rem #VECREP Private Sub S4_0%310_0%280_0%290_0%0200%040130
BOX_I:
Rem PUBIN
BOX_2:
Rem PUBKK
BOX_3:
BOX_4:
BOX_5:
BOX_6:
BOX_7:
BOX_E:
End Sub
Rem #PRCEND

```

## 【図 249】

付録一全ベクトルの型: 064

12第4のS4 (管理番号0700)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W%300_0%310_0%280_0%290_0%020 As 0%230
Rem #DEFEND
Rem #DEFPRV
Rem PUBKU
Private CNS_NOT_KUH_0%230 As 0%230
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub S4_0%310_0%280_0%290_0%0200%040140
BOX_1:
BOX_2:
Rem PUBKK
BOX_3:
BOX_4:
BOX_5:
BOX_6:
BOX_7:
BOX_E:
End Sub
Rem #PRCEND
```

## 【図 250】

付録一全ベクトルの型: 065

04第4のS4 (管理番号0710)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W%300_0%310_0%280_0%290_0%020 As 0%230
Rem PUB4N4
Private CTR_0%310_0%280_0%290_04_END As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PUBKU
Private CNS_NOT_KUH_0%230 As 0%230
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub S4_0%310_0%280_0%290_0%0200%040150
BOX_1:
BOX_2:
Rem PUBKK
BOX_3:
BOX_4:
BOX_5:
BOX_6:
BOX_7:
BOX_E:
End Sub
Rem #PRCEND
```

## 【図 2 5 1】

付録一全ベクトルの型: 0 6 6

L 4 第 4 (正規/配列無/等価無/境界無) の S 4 (管理番号 0 7 2 0)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
  Private W%30%_%31%_%28%_%29%_%02% As %23%
Rem PUB4N2
  Private CTR_%31%_%28%_%29%_04_END As Boolean
Rem #DEFEND
Rem #DEFFRV
Rem PRV7R4
  Private CTRL_W%30%_%29%_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_%23% As %23%
Rem #DEFEND
  Public Sub Main 0
Rem #VECREP Private Sub S4_%31%_%28%_%29%_%02%%04%16 0
  BOX_1:
Rem PUB04STS
  If CTR_%31%_%28%_%29%_04_END <> True Then GoTo BOX_E
  BOX_2:
Rem PUBKK
  BOX_3:
  BOX_4:
  BOX_5:
  BOX_6:
  BOX_7:
  BOX_E:
End Sub
Rem #PRCEND
```

## 【図 2 5 2】

付録一全ベクトルの型: 0 6 7

L 4 第 4 (正規/配列無/等価有/境界無) の S 4 (管理番号 0 7 3 0)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
  Private W%30%_%31%_%28%_%29%_%02% As %23%
Rem PUB4N2
  Private CTR_%31%_%28%_%29%_04_END As Boolean
Rem #DEFEND
Rem #DEFFRV
Rem PRV7R4
  Private CTRL_W%30%_%29%_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_%23% As %23%
Rem #DEFEND
  Public Sub Main 0
Rem #VECREP Private Sub S4_%31%_%28%_%29%_%02%%04%16 0
  BOX_1:
Rem PUB04STS
  If CTR_%31%_%28%_%29%_04_END <> True Then GoTo BOX_E
  BOX_2:
Rem PUBKK
  BOX_3:
  BOX_4:
  BOX_5:
  BOX_6:
  BOX_7:
  BOX_E:
End Sub
Rem #PRCEND
```



## 【図 2 5 3】

付録一全ベクトルの型: 0 6 8

L 4 第4 (正規/配列有/等価無/境界無) のS 4 (管理番号0 7 4 0)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W0%300_0%310_0%280_0%290_0%020 (0%260, 0%270) As 0%230
Rem PUB4N2
Private CTR_0%310_0%280_0%290_04_END As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV7R4
Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_0%230 As 0%230
Rem #DEFEND
Public Sub Main 0
Rem #VECREP Private Sub S4_0%310_0%280_0%290_0%0200%04016 0
BOX_1:
Rem PUB04STS
If CTR_0%310_0%280_0%290_04_END <> True Then GoTo BOX_E
BOX_2:
Rem PUBKK
BOX_3:
BOX_4:
BOX_5:
BOX_6:
BOX_7:
BOX_E:
End Sub
Rem #PRCEND
```

## 【図 2 5 4】

付録一全ベクトルの型: 0 6 9

L 4 第4 (正規/配列有/等価有/境界無) のS 4 (管理番号0 7 5 0)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W0%300_0%310_0%280_0%290_0%020 (0%260, 0%270) As 0%230
Rem PUB4N2
Private CTR_0%310_0%280_0%290_04_END As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV7R4
Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_0%230 As 0%230
Rem #DEFEND
Public Sub Main 0
Rem #VECREP Private Sub S4_0%310_0%280_0%290_0%0200%04016 0
BOX_1:
Rem PUB04STS
If CTR_0%310_0%280_0%290_04_END <> True Then GoTo BOX_E
BOX_2:
Rem PUBKK
BOX_3:
BOX_4:
BOX_5:
BOX_6:
BOX_7:
BOX_E:
End Sub
Rem #PRCEND
```

## 【図 2 5 5】

付録一全ベクトルの型: 070

L 4 第 4 (正規/配列無/等価無/境界有) の S 4 (管理番号 0760)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
  Private W0%300_0%310_0%280_0%290_0%020 As 0%230
Rem PUB4N2
  Private CTR_0%310_0%280_0%290_04_END As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV7R4
  Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_0%230 As 0%230
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub S4_0%310_0%280_0%290_0%0200%040160
BOX_1:
Rem PUB04STS
  If CTR_0%310_0%280_0%290_04_END <> True Then GoTo BOX_E
BOX_2:
Rem PUBKK
BOX_3:
BOX_4:
BOX_5:
BOX_6:
BOX_7:
BOX_E:
End Sub
Rem #PRCEND
```

## 【図 2 5 6】

付録一全ベクトルの型: 071

L 4 第 4 (正規/配列無/等価有/境界有) の S 4 (管理番号 0770)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
  Private W0%300_0%310_0%280_0%290_0%020 As 0%230
Rem PUB4N2
  Private CTR_0%310_0%280_0%290_04_END As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV7R4
  Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_0%230 As 0%230
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub S4_0%310_0%280_0%290_0%0200%040160
BOX_1:
Rem PUB04STS
  If CTR_0%310_0%280_0%290_04_END <> True Then GoTo BOX_E
BOX_2:
Rem PUBKK
BOX_3:
BOX_4:
BOX_5:
BOX_6:
BOX_7:
BOX_E:
End Sub
Rem #PRCEND
```

## 【図 257】

付録—全ベクトルの型: 072

L4第4 (正規/配列有/等価無/境界有) のS4 (管理番号0780)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W0%300_0%310_0%280_0%290_0%020 (0%260, 0%270) As 0%230
Rem PUB4N2
Private CTR_0%310_0%280_0%290_04_END As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV7R4
Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_0%230 As 0%230
Rem #DEFEND
Public Sub Main 0
Rem #VECREP Private Sub S4_0%310_0%280_0%290_0%0200%04016 0
BOX_1:
Rem PUB04STS
If CTR_0%310_0%280_0%290_04_END <> True Then GoTo BOX_E
BOX_2:
Rem PUBKK
BOX_3:
BOX_4:
BOX_5:
BOX_6:
BOX_7:
BOX_E:
End Sub
Rem #PRCEND
```

## 【図 258】

付録—全ベクトルの型: 073

L4第4 (正規/配列有/等価有/境界有) のS4 (管理番号0790)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W0%300_0%310_0%280_0%290_0%020 (0%260, 0%270) As 0%230
Rem PUB4N2
Private CTR_0%310_0%280_0%290_04_END As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV7R4
Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_0%230 As 0%230
Rem #DEFEND
Public Sub Main 0
Rem #VECREP Private Sub S4_0%310_0%280_0%290_0%0200%04016 0
BOX_1:
Rem PUB04STS
If CTR_0%310_0%280_0%290_04_END <> True Then GoTo BOX_E
BOX_2:
Rem PUBKK
BOX_3:
BOX_4:
BOX_5:
BOX_6:
BOX_7:
BOX_E:
End Sub
Rem #PRCEND
```

## 【図 259】

付録一全ベクトルの型: 074

L4第4 (K/配列無/等価無/境界無) のS4 (管理番号0800)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
  Private W0%300_0%310_0%280_0%290_0%020 As 0%230
Rem PUB4N2
  Private CTR_0%310_0%280_0%290_04_END As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV7R4
  Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_0%230 As 0%230
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub S4_0%310_0%280_0%290_0%020040160
  BOX_1:
Rem PUB04STS
  If CTR_0%310_0%280_0%290_04_END <> True Then GoTo BOX_E
  BOX_2:
Rem PUBKK
  BOX_3:
  BOX_4:
  BOX_5:
  BOX_6:
  BOX_7:
  BOX_E:
End Sub
Rem #PRCEND
```

## 【図 260】

付録一全ベクトルの型: 075

L4第4 (K/配列無/等価有/境界無) のS4 (管理番号0810)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
  Private W0%300_0%310_0%280_0%290_0%020 As 0%230
Rem PUB4N2
  Private CTR_0%310_0%280_0%290_04_END As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV7R4
  Private CTRL_W0%300_0%290_ING_FLG As Boolean
Rem PUBKU
  Private CNS_NOT_KUH_0%230 As 0%230
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub S4_0%310_0%280_0%290_0%020040160
  BOX_1:
Rem PUB04STS
  If CTR_0%310_0%280_0%290_04_END <> True Then GoTo BOX_E
  BOX_2:
Rem PUBKK
  BOX_3:
  BOX_4:
  BOX_5:
  BOX_6:
  BOX_7:
  BOX_E:
End Sub
Rem #PRCEND
```

## 【図 261】

付録—全ベクトルの型: 076

L 4 第4 (K/配列有/等価無/境界無) のS4 (管理番号0820)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W%300_0%310_0%280_0%290_0%020 (0%260, 0%270) As 0%230
Rem PUB4N2
Private CTR_0%310_0%280_0%290_04_END As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV7R4
Private CTRL_W%300_0%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_0%230 As 0%230
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub S4_0%310_0%280_0%290_0%0200%040160
BOX_1:
Rem PUB04STS
If CTR_0%310_0%280_0%290_04_END <> True Then GoTo BOX_E
BOX_2:
Rem PUBKK
BOX_3:
BOX_4:
BOX_5:
BOX_6:
BOX_7:
BOX_E:
End Sub
Rem #PRCEND
```

## 【図 262】

付録—全ベクトルの型: 077

L 4 第4 (K/配列有/等価有/境界無) のS4 (管理番号0830)

```
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private W%300_0%310_0%280_0%290_0%020 (0%260, 0%270) As 0%230
Rem PUB4N2
Private CTR_0%310_0%280_0%290_04_END As Boolean
Rem #DEFEND
Rem #DEFPRV
Rem PRV7R4
Private CTRL_W%300_0%290_ING_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_0%230 As 0%230
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub S4_0%310_0%280_0%290_0%0200%040160
BOX_1:
Rem PUB04STS
If CTR_0%310_0%280_0%290_04_END <> True Then GoTo BOX_E
BOX_2:
Rem PUBKK
BOX_3:
BOX_4:
BOX_5:
BOX_6:
BOX_7:
BOX_E:
End Sub
Rem #PRCEND
```

## 【図 2 6 3】

付録一全ベクトルの型: 0 7 8

L 2 第4 (配列無/等価無/境界無) の S 4 (管理番号 0 8 8 0)

```

Option Explicit
Rem *** VECDEF
Rem #DEFPUB
Rem PUB4N4
Private W0%300_0%310_0%280_0%290_0%020 As 0%230
Rem #DEFEND
Rem #DEFPRV
Rem PUBPKY04
Private CTRL_W0%300_0%310_0%280_0%290_0%020_OUT_Conditions_KEY_NOT_VALID_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_0%230 As 0%230
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub S4_0%310_0%280_0%290_0%0200%040170
BOX_1:
Rem PUBPKY04
If CTRL_W0%300_0%310_0%280_0%290_OUT_Conditions_KEY_NOT_VALID_FLG = True Then GoTo BOX_E
BOX_2:
Rem PUBKK
BOX_3:
BOX_4:
BOX_5:
BOX_6:
BOX_7:
BOX_E:
End Sub
Rem #PRCEND

```

## 【図 2 6 4】

付録一全ベクトルの型: 0 7 9

L 2 第4 (配列有/等価無/境界無) の S 4 (管理番号 0 8 9 0)

```

Option Explicit
Rem *** VECDEF
Rem #DEFPUB
Rem PUB4N4
Private W0%300_0%310_0%280_0%290_0%020 As 0%230
Rem #DEFEND
Rem #DEFPUB
Rem PUBPKY04
Private CTRL_W0%300_0%310_0%280_0%290_0%020_OUT_Conditions_KEY_NOT_VALID_FLG As Boolean
Rem PUBKU
Private CNS_NOT_KUH_0%230 As 0%230
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub S4_0%310_0%280_0%290_0%0200%040170
BOX_1:
Rem PUBPKY04
If CTRL_W0%300_0%310_0%280_0%290_OUT_Conditions_KEY_NOT_VALID_FLG = True
Then GoTo BOX_E
BOX_2:
Rem PUBKK
BOX_3:
BOX_4:
BOX_5:
BOX_6:
BOX_7:
BOX_E:
End Sub
Rem #PRCEND

```

【図 265】

付録-全ベクトルの型: 080

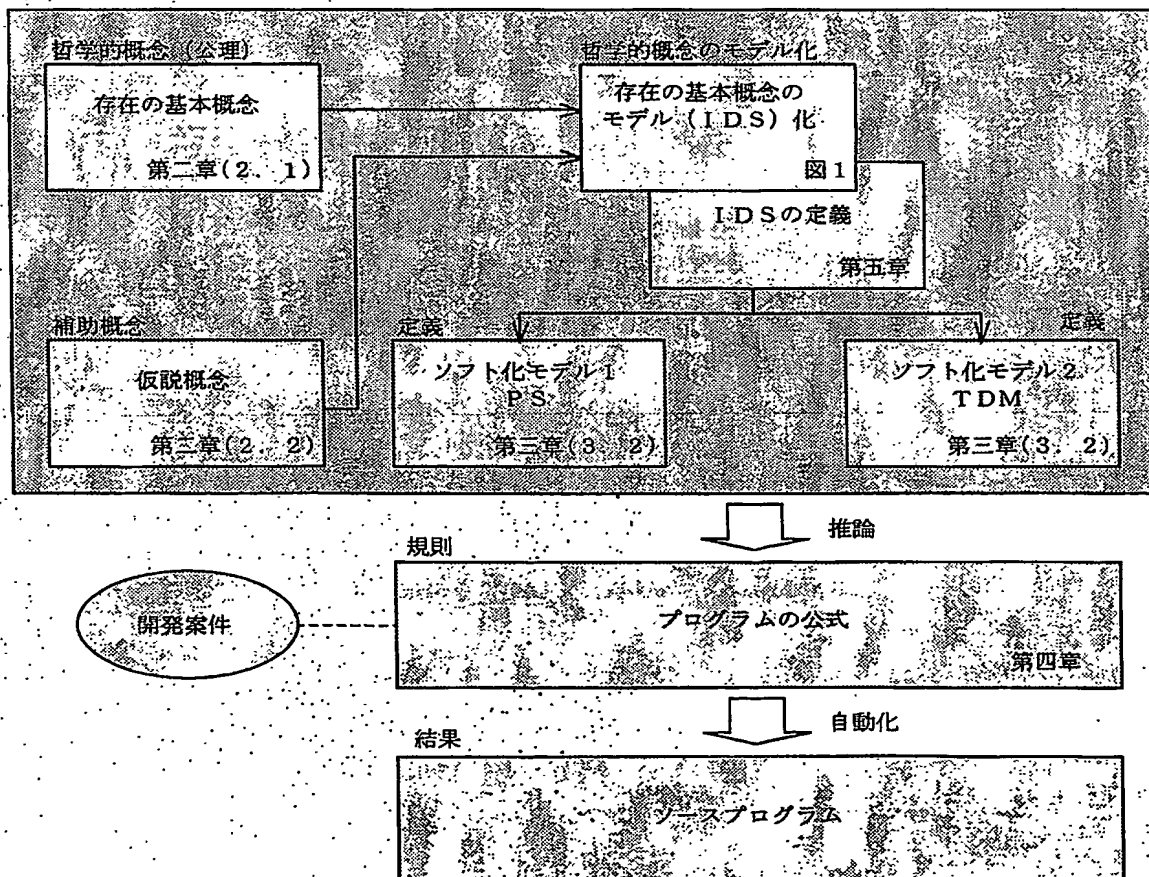
L3第4 (正規/配列無/等価無/境界無) のS4 (管理番号0900)

```

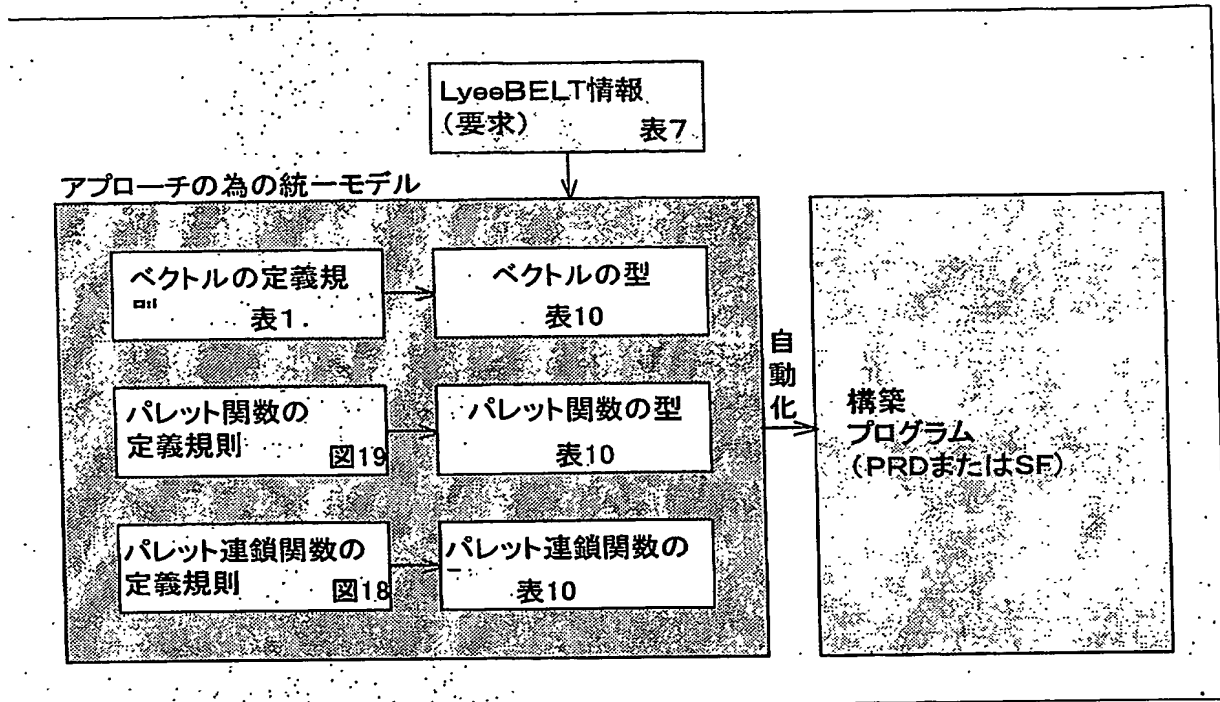
Option Explicit
Rem #DEFPUB
Rem PUB4N4
Private R0%300_0%316_0%280_0%290_0%020 As 0%230
Rem #DEFEND
Rem #DEFPRV
Rem PUBKU
Private CNS_NOT_KUH_0%230 As 0%230
Rem #DEFEND
Public Sub Main0
Rem #VECREP Private Sub S4_0%310_0%280_0%290_0%0200%040180
BOX_1:
BOX_2:
Rem PUBKK
BOX_3:
BOX_4:
BOX_5:
BOX_6:
BOX_7:
BOX_E:
End Sub
Rem #PRCEND

```

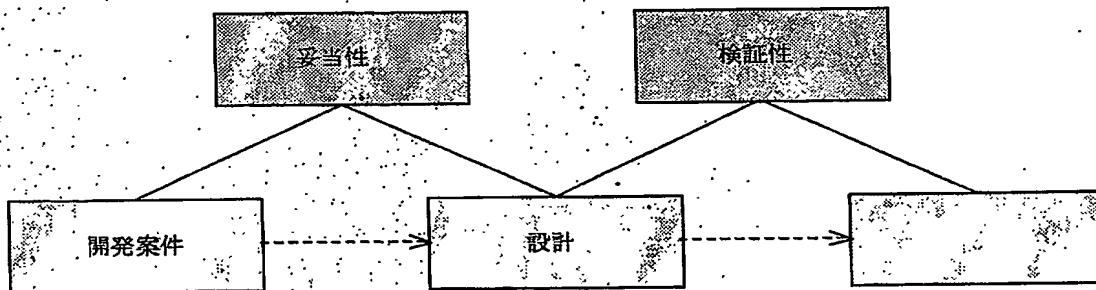
【図 266】



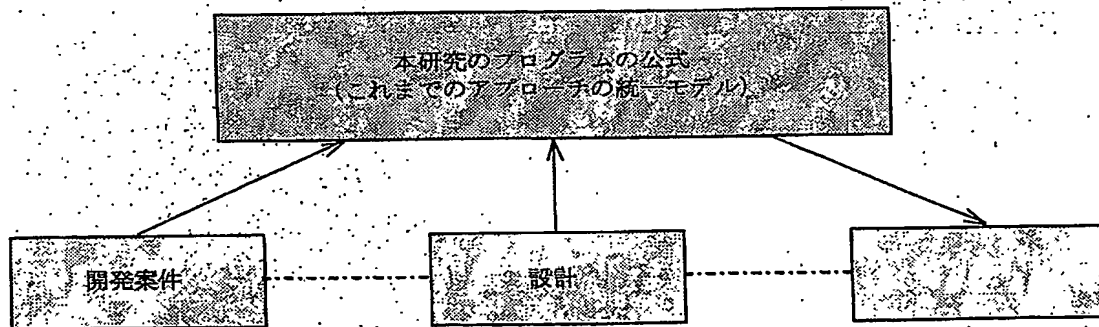
【図 267】



【図 268】



【図 269】





【書類名】 要約書

【要約】

【課題】

ソフトをプログラム言語で述定する為に開発案件をアルゴリズム化する標準規則（公式）を求める事。

【解決手段】

例えば、われわれが言葉を発する直前迄、発する言葉を記憶する事が出来ないにも拘らず、われわれは言葉が無意識的に紡いで文章にしている。本研究では、これと同じ原理でプログラムが述定出来ないだろうかとして論考される。

この為に、本研究では「存在」「意図」「意識」に関する論考を形而上学的に行う。其して、其の結果、ソフトは「意識を成立させる外延的な存在を内包する作用」として定義される。其して、この定義から誘導される新たな定義により、プログラムを定義する公式が求められる。

【選択図】 図 19

**【書類名】** 出願人名義変更届  
**【提出日】** 平成16年 8月 2日  
**【あて先】** 特許庁長官 殿  
**【事件の表示】**  
**【出願番号】** 特願2003-330797  
**【承継人】**  
**【識別番号】** 396023362  
**【住所又は居所】** 東京都江東区潮見 2 丁目 1 0 番 2 4 号  
**【氏名又は名称】** カテナ株式会社  
**【代表者】** 小宮 善継  
**【提出物件の目録】**  
**【物件名】** 権利の承継を証明する書面 2

【物件名】

権利の承継を証明する書面

【添付書類】

ニ  2004

平成 16 年 7 月 30 日

持 分 譲 渡 証 書

住 所 東京都江東区潮見 2 丁目 10 番 24 号  
譲受人 カ テ ナ 株 式 会 社  
代表取締役 小 宮 善 継 殿

住 所 東京都港区高輪 3 丁目 11 番 3 号  
譲渡人 ソフトウェア生産技術研究所株式会社  
代表取締役 小 宮 善 継



下記の発明に関する特許を受ける権利については弊社と  
株式会社アイエスデー研究所との共有のところ、今般、  
弊社の持分を貴社に譲渡したことに相違ありません。

記

1. 特許出願番号 特願 2003-330797
2. 発明の名称 ソフトウェアの生産方法

平成16年7月30日

持 分 譲 渡 証 書

住 所 東京都江東区潮見2丁目10番24号

譲受人 カ テ ナ 株 式 会 社

代表取締役 小 宮 善 継 殿

住 所 東京都港区高輪3丁目11番3号

譲渡人 株式会社アイエスデー研究所

代表取締役 根 来 文 生



下記の発明に関する特許を受ける権利については弊社と  
ソフトウェア生産技術研究所株式会社との共有のところ、  
今般、弊社の持分を貴社に譲渡したことに相違ありません。

記

1. 特許出願番号 特願2003-330797

2. 発明の名称 ソフトウェアの生産方法

認定・付加情報

特許出願の番号	特願2003-330797
受付番号	10401440020
書類名	出願人名義変更届
担当官	小暮 千代子 6390
作成日	平成16年 9月 7日

<認定情報・付加情報>

【提出された物件の記事】

【提出物件名】 権利の承継を証明する書面 1



特願 2 0 0 3 - 3 3 0 7 9 7

出 願 人 履 歴 情 報

識別番号

[ 5 9 8 1 5 3 4 0 1 ]

1. 変更年月日

1 9 9 8 年 1 0 月 2 日

[変更理由]

新規登録

住 所

東京都港区高輪三丁目 1 1 番 3 号

氏 名

株式会社アイエスデー研究所

特願 2 0 0 3 - 3 3 0 7 9 7

出 願 人 履 歴 情 報

識別番号

[ 5 9 9 0 8 6 2 3 8 ]

1. 変更年月日

1 9 9 9 年 6 月 2 1 日

[変更理由]

新規登録

住 所

東京都港区高輪 3 丁目 1 1 番 3 号

氏 名

ソフトウェア生産技術研究所株式会社

特願 2 0 0 3 - 3 3 0 7 9 7

出 願 人 履 歴 情 報

識別番号

[ 3 9 6 0 2 3 3 6 2 ]

1. 変更年月日

1 9 9 6 年 1 0 月 1 6 日

[変更理由]

新規登録

住 所

東京都江東区潮見二丁目 1 0 番 2 4 号

氏 名

カテナ株式会社



**This Page is Inserted by IFW Indexing and Scanning  
Operations and is not part of the Official Record**

**BEST AVAILABLE IMAGES**

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☒ **BLACK BORDERS**
- ☐ **IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- ☒ **FADED TEXT OR DRAWING**
- ☐ **BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- ☐ **SKEWED/SLANTED IMAGES**
- ☐ **COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- ☐ **GRAY SCALE DOCUMENTS**
- ☒ **LINES OR MARKS ON ORIGINAL DOCUMENT**
- ☐ **REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- ☐ **OTHER: \_\_\_\_\_**

**IMAGES ARE BEST AVAILABLE COPY.**

**As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.**